Z-MUSIC システムver.3.0

Oh!X Books

元Oh!X編集部編

X68000用音楽ドライバZMSC3.X
AD PCMシーケンスドライバMPCM.X
X-BASIC用外部関数/C言語用ライブラリ
X68000用各種フリーソフトウェア収録
特別付録　5"2HD2枚+CD-ROM

SOFT BANK



X68000
PERSONAL WORKSTATION

Z-MUSIC SYSTEM ver. 3.0 MANUAL

まえがき

X680x0をパーソナルワークステーションからミュージックワークステーションへ発展させよう、少々大げさですが、これが今回の開発のキーワードでした。

ver.3.0になって、もっとも大きく進歩した点は、なんといってもMIDIの最大４系統出力とAD PCMのメロディシーケンス機能です。前者はMIDIボード(CZ-6BM1/互換品)を最大２枚、そしてRS-232C MIDI２系統(PC-98用シリアルMIDIケーブル使用時)を同時に制御できる機能であり、最大64チャンネルのMIDI出力を同時に行えます。後者は(C)Wachoman氏に制作していただいたMPCM.Xが完成して初めてできるようになった機能です。彼には今回の開発で本当にいろいろと助けていただきました(この場を借りて感謝の意を表します)。MPCM.XによりX680x0の内蔵AD PCM音源をあたかもFM音源のような柔軟さで制御することができ、本来のAD PCM音源のスペックを遙かに超越した音楽表現が行えるようになりました。

X680x0は確かに現在では主流のパソコンとはいえなくなってきましたが、ことに音楽情報を扱うということにおいては、割り込みアーキテクチャの完成度の観点からは現時点の標準的なWindows95パソコンに優っています。パソコンをひとつの道具としてとらえた場合、X680x0はまだまだその道具としての役割を果たせます。

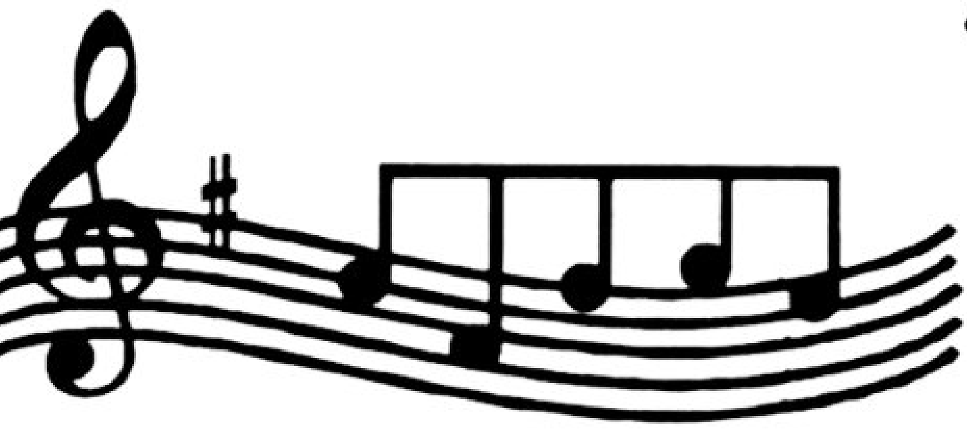
Z-MUSIC ver.3.0が、その道具としての新しい機能のひとつとなることを期待しています。

contents

MEASURE 1

イントロダクション

ここではZ-MUSICシステムの概要を説明します。



1．1． はじめに

コンピュータミュージックを楽しむにはいくつかの方法があります。
1) 実際に弾いたものをコンピュータ、シーケンサのメモリに記憶しておくもの
2) コンピュータ、シーケンサのメモリに直接演奏データを書き込んでいくもの
3) 1)と2)の複合的な方法
などです。2)はコンピュータのキーボードを叩くことから「打ち込み」と呼ばれます。3)はつまり、人間が弾いたものをあとでコンピュータを用いて修正するといったものです。

ちょっと昔までは3)の方法が圧倒的にメジャーでしたが、音源の進歩とミュージックツールの進化により最近は2)の方法もメジャー化してきています。最近は譜面も起こさずに直接「打ち込み」を始める作曲者も少なくないようですし、スタジオのマニピュレータ(平たくいえば打ち込み人)のなかには実際に弾いたものよりも人間臭さを表現する人もいると聞きます。

さて、本書でお話しするのは2)の方法です。2)の方法の魅力といえば、自分が楽器の演奏ができなくてもイメージ通りの演奏が可能という点でしょう。さらに、一度仕上げた曲は、ディスクなどの記憶装置に保存しておけばいつでも再現可能というのもこの方法の魅力です。

打ち込みの方法にも実はいろいろあります。いちばん原始的なものは完全な数値入力です。音階、発音時間、などすべてをこまごまと入力していくものです。最近は「ステップ入力」と呼ばれます。10数年前にMZやPC-8001用に出ていたローランドCMU800シリーズはこの方法でした。これを進化させものが譜面入力による「打ち込み」です。いわゆる五線譜に音符を置いていくタイプのものです。楽器の演奏はおろか譜面を読むことができない人にでも簡単に音楽を作れるため初心者層には圧倒的な人気を誇ります。また印刷機能が充実していれば美しい譜面を得ることができますので、プロでもこのタイプを使っている人が多くいます。

さて、「Z-MUSIC」では現在（注）MMLという「打ち込み」方式を採用しています。これはいわば「音楽記述言語」というもので「音楽をプログラム」するというイメージが正しいでしょう。完全な数値入力よりはずっとドキュメント性が高いうえ、入力する量も少ないのでパソコンを基盤として音楽を楽しむ人にはピッタリのものといえます。欠点としては、小中学校程度の音楽知識が必要なこと、ある程度コンピュータを扱える人でないと駄目なこと、ドキュメント性が譜面入力よりは低いこと、などです。ドレミが読めて、非GUI環境でディスクをフォーマットしたりファイルをコピーしたことがあれば資質十分です。

MMLの利点としては、慣れればあらゆる方式よりも高速に１曲を仕上げることができ、細かいニュアンスなどに凝ることができるなどが挙げられます。また、移植性が高い点も見逃せません。ほとんど方言のような感覚で他機種の演奏データやほかの音楽ドライバ用のデータを、自分の使っているMMLに修正利用することができるのです。

さあ、「Z-MUSIC」を懸け橋にしてあなたもコンピュータミュージックの世界に浸りませんか。

(注) 今後、違った入力方式に対応した支援プログラムが発表されれば、別の入力方式（たとえば譜面入力方式）も可能です。

1．2． Z-MUSICとは

Z-MUSICとはX680x0本体付属のOPMDRV.Xを改造したりするものではなく、まったく新しくゼロから開発されたミュージックプログラム開発言語です。X680x0の内蔵音源であるFM音源８声とAD PCM１声（MPCM.X(C)Wachomanを使用すれば16声まで）、MIDIボードが接続されていればMIDI楽器も同時にコントロールすることができます。

一般の音楽制御ドライバでは１台のMIDI楽器をコンピュータの外部音源という位置付けで扱っていましたが「Z-MUSIC」ではX680x0をホストに複数のMIDI楽器をコントロールすることができるように設計されているのでどんどん自分のMIDIシステムを広げていくことが可能です。

特に今回のver.3.0は多チャンネル化するMIDI音源に対応する目的と、いっそうのコンピュータ音楽ならではの芸術性や表現を具現化するために制作されました。より高度なコンピュータ音楽を制作できるはずです。

ここでZ-MUSICシステムver.3.0の特長を列挙してみます。
1) X680x0本体付属のOPMDRV.Xに上位コンパチ。Z-MUSIC ver.2.0以前の演奏データはソース(ZMS)レベルで上位互換でそのまま演奏可能
2) FM音源，AD PCM，MIDI楽器を同時に同期演奏可能(もちろんMIDIシステムのない環境でもZ-MUSICは利用可能)
3) 複数のX680x0を相互に接続しての同期演奏も可能
4) 汎用トラックを65535本装備。最大同時演奏トラック数65535本。RS-232C MIDIを2ポートと2枚のMIDIボードを同時に制御可能。MIDIは、前バージョンの４倍に当たる64チャンネルまでを同時制御可能。最大同時演奏チャンネル数88チャンネル(FM8+AD PCM16+MIDI64)
5) 独自のAD PCMドライバ「MPCM.X」(C)WachomanにてAD PCM音源をFM音源のように柔軟に制御可能
6) ポルタメントやオートベンド、和音やビブラートなどの特殊効果を内蔵音源とMIDIの両方で使用可能。ver.2.0以前からあったARCC機能も、１トラックあたり同時に４つまで独立に動作させることができるように拡張された
7) コンピュータ音楽ならではの音楽情緒を作り出すアゴーギク機能やエンハンスドベロシティシーケンス機能の新設
8) 前バージョンまでの効果音モードを発展させ、２曲までテンポの異なった音楽を同時に演奏可能
9) 送信MIDIデータをリアルタイムで最適な送信方法を選択して送信するVTMS機構、送信MIDIメッセージをリアルタイムに最適化するARS機構を装備。これらにより多チャンネル演奏時でもテンポずれの最大限抑止、CPU負荷の低減を両立
10) MT-32/U220/M1/SC-55/SC-88/GM音源などに対応した楽器個別の制御命令を装備
11) MIDI楽器側の音色や設定データ、ユーザーの演奏をスタンダードMIDIファイルに出力可能
12) オブジェクトレベルの演奏データ(ZMD)を出力可能（コンパイル機能）
13) Z-MUSICが持つ機能のすべてを公開。外部プログラムから利用可能。プログラム間通信機能も装備し、高度な連動動作機能を提供
14) 専用A/Dコンバータによってサンプリングされた高音質のAD PCMデータライブラリを標準装備
15) 全情報公開。ライセンスフリー

1．3． 著作権について

法律上、日本では著作権の放棄ができませんので、著作権は作者西川善司に保留されます。しかし、プログラムの性質上、「Z-MUSIC」のオリジナルを開発した私、西川善司は「Z-MUSIC」およびこれらを支援するプログラム(サブルーチンを含む)すべての使用権に関するライセンス権を放棄します。よって特に断らずに商的利用ができます。つまり市販だろうが同人だろうが勝手に「ZMSC3.X」を組み込んだソフトを販売してもいいとい

うことです。
ただし、ひとつだけ守ってほしいことがあります。それは「Z-MUSIC」の改造についてです。改造はもちろん各自で自由で行って構わないのですが、データの完全な互換性が保持されないような改造を施した際にはバージョンID番号を$F0以上にしてほしいのです(オリジナルは$30です。ソースリストZMSC0.HAS参照)。また、オリジナルのZ-MUSICで演奏できないデータフォーマットにした場合はファイルの拡張子もZMD,ZMS以外に変更してください。これは混乱を防止するためです。今後のバージョンアップ版との兼ね合いもありますので、ぜひご協力ください。
移植に関しても自由に行って構いません(ただし、完全にコンパチでない場合はバージョン番号をやはり$F0以上にしてください)。

1．4．　ZMDとZMS

Z-MUSICシステム上での作業方法や機能の仕組みについて簡単に解説します。
Z-MUSICシステムver.3.0がver.2.0以前と大きく異なる点はMMLコンパイラと演奏プログラムを分離したところにあります。ver.3.0ではコンパイラは必要なときに呼び出され、あるいは、ユーザーの意思の下に常駐させることができ、ver.2.0以前に比べて省メモリを実現しています。
Z-MUSICの機能を直接的に言語命令化したものがZ-MUSICのMMLですが、Z-MUSICの機能を別な形式で結びつければZ-MUSICを活用したMML以外の音楽制作環境の実現も可能です。または、独自の文法のMMLコンパイラを制作すれば、見た目まったく別の音楽データを演奏させることもできるのです。これがコンパイラ分離のもうひとつの狙いであり利点なのです。
Z-MUSICを利用するには、その演奏プログラムを常駐させなければなりません。具体的には、

A>ZMSC3　　[リターン]

のようにします。ZMSC3.Xは、Z-MUSICシステムでは演奏マネージャと呼ばれ音楽の実際の演奏やZ-MUSICの機能の提供を司ります。ZMSC3.Xを常駐させることによってはじめてZ-MUSICシステムの機能が使えるようになります。
次に音楽データの制作手法ですが、現時点でZ-MUSIC ver.3.0で演奏(音楽)データを制作するには、X680x0本体付属のED.Xのようななんらかのテキストエディタが必要です。テキストエディタでMEASURE 5で解説されているようなMML(音楽記述言語)を駆使して音楽プログラムを作成します。MMLにて制作した音楽プログラムをZ-MUSICでは特にZMSと呼んでいます。
これを演奏させるには、

A>ZP3 ファイル名.ZMS

のようにします。ZP3.Rは演奏データとZMSC3.Xの間を取り持つ役目を果たします。ZP3.Rは、ZP3.Rに与えるファイルがZMSの場合、MMLコンパイラZMC.Xを起動します。これでコンパイルを行い、音楽記述言語レベルからZMSC3.Xで実際に演奏ができるバイナリファイルに変換します。このバイナリファイルをZ-MUSICシステムではZMDと呼んでいます。
ZMC.XでZMSをあらかじめZMDに変換して保存しておくこともできます。
具体的には、

A>ZMC ファイルネーム.ZMS

のようにします。エラーがなければZMDが作成されるはずです。このZMDがディスク上にあるのならば、

A>ZP3 ファイルネーム.ZMD

として演奏させることもでき、この場合はコンパイラを呼び出したり、コンパイル動作などのプロセスが省略されるため即演奏が開始されます。ただし、ZMDは仕様改良にともなって変更される場合があるためZMSは必ず保存しておくようにしてください。ZMDの仕様変更はありえますが、ZMSレベルでは上位互換を保証していきます。よって通信やその他のメディアで制作した音楽を公開する場合にも、必ずZMSを添付するようにしてください。
ただし、冒頭で述べた独自コンパイラや独自音楽制作ツールで制作した音楽データの場合はこの限りではありません。しかしそういったコンパイラやツールにおいてもZMD仕様変更などに対応できるようにかなり柔軟な設計にしておくべきでしょう。

1．5．　その他のZ-MUSIC関連ファイル

ZMDとZMS以外にZ-MUSICが取り扱うファイルには、

1)　OPMファイル
2)　ZDFファイル
3)　CNFファイル
4)　ZPDファイル
5)　MIDファイル
6)　MDDファイル
7)　JUKファイル

があります。
1)はX680x0付属のFM音源ドライバOPMDRV.X用の音楽データです。Z-MUSICはOPMDRV.Xの上位互換であるため、OPMDRV.Xの演奏データを演奏させることができます。ただしZ-MUSICはエラーチェックが厳格であるため、OPMDRV.Xでは演奏できてもZ-MUSICではエラーとなる場合があります。その場合はエラーメッセージを参考に演奏データをZ-MUSICの文法チェックをパスするように修正しなければなりません。
2)はLZZという名の音楽データの圧縮ツールによって圧縮された音楽データです。このLZZはMZP.XなどのZ-MUSIC関連ツールを数多く制作されているNOVA氏の作品で、Z-MUSICにも対応しておりZMS,ZMD,ZPD,MDDなどを圧縮することができます。Z-MUSICシステム ver.3.0ではZP3.XがこのZDFをデコードして演奏/再生することができます。
3)、4)はZ-MUSIC上でAD PCM音を扱う場合に深く関係するファイルです。3)はユーザーが使用する任意のAD PCMデータを選択しリスト化したファイルです。4)は3)ファイルからAD PCMファイルコンバータZPCNV3.Rで作成したAD PCMデータのアーカイブファイルです。
5)はスタンダードMIDIファイル(SMF)です。Z-MUSIC ver.3.0ではフォーマット0のSMFを再生することができます。また、レコーディング機能によって記録した演奏、もしくは楽器などの設定ダンプデータもSMFに変換することができます。
6)はZ-MUSIC ver.2.0以前で扱っていたMIDIダンプデータと呼ばれる楽器などの設定をファイル化したテキスト型MIDIデータファイルです。ver.3.0ではこちらは再生のみ行えます。
7)は、ZP3.Rのジュークボックス機能で演奏させる曲目を列挙したリストファイルです。

1．6．　本マニュアルについて

本マニュアルではZ-MUSIC ver.3.0で新設された機能や拡張された機能を中心に取り扱います。ver.2.0との上位互換のためだけに残っている機能については特に解説しませんのでご了承ください。具体的には、

・MUSICZ.FNC関数命令群
(ただしver.3のX-BASIC外部関数であるMUSICZ3.FNCについては詳しく解説します)
・MDDファイル
(スタンダードMIDIファイル(SMF)を中心的に取り扱うことになったため)

1．7．　付属フロッピーディスクとCD-ROMについて

付属のフロッピーディスク１をドライブ０に、フロッピーディスク２をドライブ１に入れて[OPT.1]キーを押しながらX680x0を起動するとディスクマガジンスタイルのZ-MUSIC Ver.3.0のクイックスタートマニュアルが始まります。Z-MUSICシステムの具体的なインストール方法や、X680x0とMIDI機器との接続方法など、本マニュアルに記載されていないような情報が掲載されています(さらにサンプル曲の演奏なども楽しめます)。実際のインストールの前に一度、目を通すことをおすすめします。
Z-MUSICシステムのプログラム類自体は付属フロッピーディスク1の中のディレクトリ「ZMSC3」、あるいは、付属CDROMの中のディレクトリ「ZM3MSTR」中(MPCM.Xは「X_USERS¥MPCM」)に収録されています。なお、ソースリストおよびマニュアルドキュメントはCDROMのほうのみにしか収録していません。
CD-ROMは、Z-MUSIC ver.2.0のムックに付属してきた6枚のフロッピーディスクをディレクトリ「ZM2DISK1～ZM2DISK6」に、最終版のZ-MUSIC ver.2.0であるver.2.08をディレクトリ「X_USRES¥ZMUSICV2」に収録しています。ver.3.0とあわせて活用してください。
そしてCD-ROMには、Z-MUSICシステム関連のもの以外に、100種以上に及ぶフリーソフト群、2,000種類以上の(AD)PCMデータなどを収録しています。各フリーソフトの概要と一覧はCD-ROMのディレクトリ「X_UERS」に含まれるFSW_LIST.TXTに記載されています。各フリーソフトの利用方法については各ソフトのアーカイブ中に含まれるドキュメント類を参照してください(使用方法、内容に関する問い合わせなどは各ソフトの作者にお願いします)。
収録(AD)PCMデータは拡張子によって異なったデータフォーマットになっています。拡張子「P16」「PCM」の利用方法については本マニュアルMEASURE6を、「M44」「M30」については各データの収録ディレクトリにあるドキュメントを参照してください。

MEASURE 2

ZMSC3.X/ZMC.Xのオプションスイッチ

ここではZ-MUSICプログラムのコマンドオプションについて解説します。

2．1． はじめに

Z-MUSICシステム ver.2.0の中核であった音楽制御ドライバZMUSIC.Xは今回のver.3.0へのバージョンアップにともない、音楽演奏制御ドライバとMMLコンパイラに分離された。

音楽演奏制御ドライバだけではZMDしか演奏することができないが、音楽演奏制御ドライバを組み込んだうえでMMLコンパイラを組み込むと音楽演奏制御ドライバにMMLコンパイル機能を持たせることができる。

コンパイラはそれ単体で実行ができるため、あらかじめZMSをコンパイラでZMDにコンパイルしておけば、コンパイラの常駐は不要となり省メモリにつながる。また、たとえ演奏したい演奏データがZMSであっても後述のZ-MUSIC ver.3.0標準プレイヤーZP3.Rを使えば自動的にコンパイラをチャイルドプロセスで呼びだしコンパイルしてからZMDの演奏を行ってくれるのでコンパイラを常駐していなくても、あたかもZMSの演奏ができるかのように演奏を行ってくれる。

本章では音楽演奏制御ドライバ「ZMSC3.X」とMMLコンパイラ「ZMC.X」のコマンドオプションについて解説する。

各スイッチは半角文字の「/」「-」の後ろにコマンドスイッチの半角アルファベットを記述し、さらに必要であればその直後に数値/文字などのパラメータを記述することによって指定する。

各スイッチは(例外を除いて)順不同でいくつでも記述することができる。

2．2． ZMSC3.Xのコマンドオプション

[注意]

説明の都合上、見出しのコマンドオプションは全角英字になっているが実際は半角(大文字/小文字いずれも可)を用いて指定する。

同じく説明の都合上、コマンドオプションのパラメータは英字小文字で表記してある。

－2

ZMSC3.Xの動作をZMUSIC.X ver.2.0xの動作に近づける。

ver.2.0とver.3.0では機能拡張にともない以下の仕様変更がなされている。

・波形メモリの波形に対して振幅を与えることができる。(ver.2.0では原波形の再生のみ)

・PCMの周波数切り換えMMLコマンド@F5,@F6が別の周波数(MEASURE5参照)に割り当てられた(ver.2.0では@F5が16bitPCM,@F6が8bitPCMを表す)。

・絶対音長1の音符は通常の音符として扱われる(ver.2.0では絶対音長1の音符はいかなるときもタイが添付されて処理される)。

本オプションを設定すると以上の点の動作をver.2.0と同じように実行するようになる。通常は設定する必要はない。

－A

音楽演奏割り込みにFM音源タイマAを使用する。デフォルトでは、MIDIボード装着時はMIDIタイマ(YM3802タイマ)を利用するが、未装着時はFM音源タイマBを使用する。

FM音源タイマAはマスタークロック(*1)192時、テンポ77から300までを有効範囲とし、かなり正確なテンポをキープできる。外部シーケンサ/リズムマシンとの同期演奏にはこちらのモードを奨励する。

よくわからない人は特に設定する必要はない。

(*1)マスタークロックについてはMEASURE3 zm_set_master_clock(), MEASURE4 .MASTER_CLOCKを参照。

－B

音楽演奏割り込みにFM音源タイマBを使用する。デフォルトでは、MIDIボード装着時はMIDIタイマ(YM3802タイマ)を利用するが、未装着時はFM音源タイマBを使用する。

FM音源タイマBはマスタークロック(*1)192時、テンポ20から32767までを有効範囲とする。

よくわからない人は特に設定する必要はない。

－E

外部シーケンサ/リズムマシンとの同期演奏を行う。X680x0をホストにしたミュージックシステムならばこのスイッチは設定する必要はない。複数のX680x0を同期させて演奏させるにはこのスイッチを設定すること。

具体的には、演奏開始時にスタートメッセージ$FA、演奏再開時にコンティニューメッセージ$FB、演奏停止時にストップメッセージ$FCを送信するようになる。また、演奏データ中で指定したテンポに合わせてタイミングクロック$F8を送信する。

－Fn

効果音モード用にトラックをn本確保する。効果音モードを利用するときはnに1～65534を設定する。デフォルトではn=0で「効果音モードを利用しない」設定となる。効果音トラックに何トラック割り当てても音楽演奏のトラック数の最大数には影響ない。

例

A>ZMSC3 -F4 (効果音トラックに4トラック確保)

－G

ZMSC3.X起動時のロゴ表示や常駐を報告するメッセージを表示しない。

－H

簡易ヘルプを表示する。-Hでなく、-?としても同様の簡易ヘルプを表示する。また、後述の-Jと併用して日本語簡易ヘルプを表示することもできる。

－In

MIDIのインタフェイスの配列を設定する。Z-MUSIC ver.3.0ではMIDIのインタフェイスには絶対的なIDを以下のように定めてある。

ID	インタフェイス名	
1	CZ-6BM1 1枚目	(CZ-6BM1はシャープ純正MIDIボード。
2	CZ-6BM1 2枚目	互換品も、説明の都合上これで表す)
3	RS-232C MIDI A	
4	RS-232C MIDI B	

-IオプションではこのIDを列記して、Z-MUSICシステム上で扱うインタフェイス番号との対応付けを行う。IDを書いた順番にインタフェイス番号が割り当てられていく。たとえばいちばん初めに書いたIDのインタフェイスにはインタフェイス番号1が割り当てられる。

ID=4は後述のオプション-WでデュアルRS-232C MIDIに設定しないと利用できない。

例

A>ZMSC3.X -I3412

この例ではMIDIインタフェイス番号1にRS-232C MIDI A,MIDIインタフェイス番号2にRS-232C MIDI B,MIDIインタフェイス番号3にCZ-6BM1 1枚目,M

IDIインタフェイス番号3にCZ-6BM1 2枚目を割り当てている。

－J

出力するメッセージを日本語にする。
この設定を行って常駐すると常駐後のメッセージもすべて日本語になる。
またZMSC3.Xのヘルプメッセージはデフォルトでは英語だが、

A>ZMSC3.X -J -H

とすると日本語で表示させることができる。

－N

初期化処理を簡略化する。この設定を行ってZMSC3.Xを常駐させると以後、演奏開始時に最低限の初期化処理しか行わなくなる。ゲームなどのBGMを機械語レベルで演奏制御する場合に、曲の変り目の処理を軽くすることができる。

通常は設定する必要はない。

－P

ポリフォニックプレッシャーの送信を抑制する。

ZMSC3.Xでは同一ノート番号に対して異なるベロシティで発音要求がなされるとポリフォニックプレッシャーを自動的に発信する。MMLではたとえば、

C*0,48,110 C4,4,120

とするとベロシティ110のCとCに対してポリフォニックプレッシャー120が送信される。このオプションを設定するとこのポリフォニックプレッシャーの送信を行わなくなる。

通常は設定する必要がない。

－R

常駐しているZMSC3.Xをシステムから削除する。

－S filename

スタンダードMIDIファイル(MID,SMF)、ZMDなどのセットアップファイルを読み込みZMSC3.X起動時に実行する。ファイルネームの拡張子などの省略は認められない。

ここにZMDのファイル名を書けばZMSC3.X起動時に曲を演奏させることも可能。コンパイラが現在分離してしまっている関係上、-SでZMSファイルを演奏させることはできない。

例

A>ZMSC3.X -Smusic.zmd

－W

RS-232C MIDIをデュアルRS-232C MIDIとして使用する。デュアルRS-232C MIDIモードではMIDIボード２枚分としてRS-232C MIDIを利用することができる。

COME ON MUSIC社のMA01のようなRS-232C MIDI変換アダプタを用いて接続するのではなく、Roland製RSC-15N(￥3,500)などのシリアルMIDIケーブルでX680x0のRS-232Cポートとモジュール側の「COMPUTER-IN」端子を接続して使用する。

このオプションを指定しないとシリアルMIDIケーブルで接続していたとしてもMIDI 1ポートとしてしかRS-232Cを利用できない。

－Xn

MIDIメッセージ$F7(EOX/エンド・オブ・エクスクルーシブ) メッセージを送信後、

$\frac{n}{60}$ 秒　　($0 \leq n \leq 255$)

のウェイトを与える。デフォルトは３で通常は設定する必要はないが、極端に応答の遅いMIDI楽器に対してはより大きな値を設定するとよい(U110/220,D70……など)。

なお、曲データ側でこのウェイトを自前で設定したい場合はn=0とする。

－Yn

RS-232C MIDIの転送速度をSCCの「時定数」で設定する。通常はMIDI規格の転送レートである31,250bpsが設定される(-Y3と同じ)。無改造マシンの場合これに2を与えると39062.5bpsになる。小さい数値を与えると速い転送レートになるが通常は設定する必要はない。ただしクロックアップ改造を施したマシンではこの設定を行う必要がある場合もある。

例

A>ZMSC3.X -Y2

－Z filename

filenameで与えられるADPCM BLOCK DATA「ZPD」をZMSC3.X常駐時に読み込む。

ファイル名拡張子を省略すると自動的に'.ZPD'が添付される。

例

A>ZMSC3.X -Zstandard.zpd

２．３．　ZMC.Xのコマンドオプション

［注意］

説明の都合上、見出しのコマンドオプションは全角英字になっているが実際は半角(大文字/小文字いずれも可)を用いて指定する。

同じく説明の都合上、コマンドオプションのパラメータは英字小文字で表記してある。

－C filename1 filename2

filename1で表されるZMSファイルをZMDへコンパイルしfilename2で保存する。

filename2は省略可能で省略時はfilename1に拡張子'.ZMD'を付けたファイル名で保存される。

この'-C'オプションを記述しなくてもfilename1とfilename2を続けてコマンドラインの最後に記述すれば設定されたと見なされ、コンパイルが実行される。

例

A>zmc -c music.zms　　　　(music.ZMDが出力される)
A>zmc -c music.zms ABC　　(ABC.ZMDが出力される)

－D

現在コンパイルしている行をリアルタイムで表示する。

このオプションを設定すると多少コンパイル速度が低下する。

－En

エラーがn個発生した時点でコンパイル処理を打ち切る。デフォルトではエラー発生個数に関わらず最後までコンパイルを行う設定になっている。

－G

ZMC.X起動時のロゴ表示や常駐を報告するメッセージを表示しない。

－H

簡易ヘルプを表示する。-Hでなく、-?としても同様の簡易ヘルプを表示する。また、後述の-Jと併用して日本語簡易ヘルプを表示することもできる。

－J

出力するメッセージを日本語にする。

この設定を行って常駐すると常駐後のメッセージもすべて日本語になる。
またZMC.Xのヘルプメッセージはデフォルトでは英語だが、

A>ZMC.X -J -H

とすると日本語で表示させることができる。

－L

ZMSC3.Xがすでに常駐している場合、このコマンドオプションを設定してZMC.Xを実行するとZMSC3.Xにリンク常駐し、コンパイラ機能を追加させることができる。

ZMSC3.Xにコンパイラ機能を持たせると、ZMSの演奏を、

A>COPY filename.ZMS ZMS

といったファイルコピーコマンドで実行できるようになる。また、ZP3.Rなどのプレイヤーで演奏させる場合もコンパイラを呼び出す処理が省略できるため、演奏開始までの時間を短縮させることができる。

－R

常駐しているZMC.Xをシステムから削除する。

CONFIG.SYSからの組み込みについて

Z-MUSIC ver3.0では動的にメモリを確保するシステムを導入したことにより、ZMSC3.XやZMC.XをCONFIG.SYSから、

DEVICE=

で組み込むことはできなくなりました。

－T

デフォルトではコンパイル中にエラーが発生すると、そのエラー箇所を表示するレポート機能が働くが、このコマンドオプションを設定すると、このレポート機能を省略させることができる。

－W

デフォルトではステップタイムなどの計算で誤差が発生すると、警告を促すが、このコマンドオプションを設定すると、これを省略させることができる。

２．４．　環境変数について

環境変数'zmsc3_opt'と'zmc_opt'に通常よく設定するコマンドオプションをあらかじめ設定しておくことができる。

たとえば、常にエラーメッセージなどを日本語で表示させておきたい場合はAUTOEXEC.BATなどに、

SET zmsc3_opt=-J

と設定しておけば、

A>ZMSC3.X

と実行しただけでも -J が追加された、

A>ZMSC3.X -J

として実行される。同様に環境変数'zmc_opt'はZMC.Xに対してのコマンドオプション設定を行うもの。

２．５．　MPCM.Xについて

MPCM.XはWachoman氏制作のポリフォニックAD PCMドライバでX680x0本体の改造をせずにAD PCM音を最大24和音、うち16音についてはリアルタイムに音程音量変換を可能にする(*1)。

Z-MUSIC ver.3.0ではこのMPCM.Xを事前に常駐させておくことにより、AD PCM音源をあたかもFM音源やMIDIのようなメロディ音源として扱うことができる。

ポリフォニック発音をさせる場合や、メロディなどを演奏させる場合は多大なCPUパワーを消費する。

以下にMPCM.Xが持つ代表的な機能を挙げる。

1)　音楽演奏用16チャンネル、効果音用８チャンネル、合計24音同時発音可能。

2)　音楽演奏用16チャンネルはオクターブ-1のCからオクターブ9のGまで半音の1/64単位の解像度で音程を変化させて発音させることができる。AD PCM音声に対してビブラートやポルタメントといったピッチエフェクトを掛けることも可能。

3)　音楽演奏用16チャンネルは音量を0-127(原音は64)まで変化させることができる。AD PCM音声に対してトレモロやエンベロープ変化などの音量エフェクトを掛けることも可能。

(*1)Z-MUSIC ver.2.0以前では江藤啓氏制作のPCM8.Xを標準のポリフォニックAD PCMドライバとしてきたが、Z-MUSIC ver.3.0ではこのMPCM.Xを標準のものとしている。またZ-MUSIC ver.3.0ではPCM8.Xを使用することはできない。

２．６　Z-MUSIC ver.3.0のファイルの検索プロセス

Z-MUSIC ver.3.0では、カレントよりファイルが見つからない場合は環境変数'zmusic_???'に書かれたパスに従ってファイルを検索する。

???の部分にはファイルの拡張子が対応する。たとえばZMSファイルを読み込む場合は環境変数'zmusic_ZMS'に書かれたパスを検索する。ファイル名に与えた拡張子がたとえ小文字のzmsでも参照される環境変数は'zmusic_ZMS'である点に注意。

なお、Z-MUSIC ver.3.0では一部の例外を除いてファイルを読み込むときには必ずこのルールに従うので、ハードディスク上の多くのディレクトリに分散されて必要ファイルが収められているときは、なるべく設定しておいたほうがよい。

例

set zmusic_ZPD=a:¥drums;a:¥pcmfiles;b:¥rythm¥snare;

↑小文字のみ可

上の例ではZPDファイルを探す場合に、a:¥drums　a:¥pcmfiles　b:¥rythm¥snareの3つのディレクトリを順番に検索するようになる。

なお、環境変数はHuman68kの仕様から255文字を超えて設定することはできない。255文字を超えて検索パスを記述したい場合には、環境変数'zmusic_???0'～'zmusic_???9'に分けて設定する。このとき、環境変数'zmusic_???'は木でいう幹に、環境変数'zmusic_???0'～'zmusic_???9'は枝の扱いになる。幹から枝への参照は、環境変数'zmusic_???'中に'/'とその後ろに補助環境変数'zmusic_???0'～'zmusic_???9'までを表す数値'0'～'9'を書き、さらにその後ろにベースとなるパスを書くことによって指定できる。

今ここでZ-MUSIC ver.3.0標準のADPCMライブラリのディレクトリ名であるBASS, SNARE, TOMTOM, CYMBAL, ETHNIC, EFFECTS, ACCENT, ORCH, RAP, ADDITIONを検索させる場合を考えるとする。各ファイルのファイル名拡張子は'PCM'であるため設定する環境変数は'zmusic_PCM'および、'zmusic_PCM0'～'zmusic_PCM9'ということになる。

たとえば、今、これらのディレクトリが、ＦドライブのADPCM_LIBの下に存在するとき、

SET zmusic_PCM=/0 F:¥ADPCM_LIB;

↑↑zmusic_PCM0を指し示す数値

補助環境変数の検索を指示するコマンド記号'/'。('-'で代用も可能)

SET zmusic_PCM0=BASS;SNARE;TOMTOM;CYMBAL;ETHNIC;EFFECTS;ACCENT;ORCH;RAP;ADDITION

と指定すればよいことになる。なお、この設定で、

F:¥ADPCM_LIB¥BASS,　F:¥ADPCM_LIB¥SNARE, F:¥ADPCM_LIB¥TOMTOM,
F:¥ADPCM_LIB¥CYMBAL, F:¥ADPCM_LIB¥ETHNIC,F:¥ADPCM_LIB¥EFFECTS,
F:¥ADPCM_LIB¥ACCENT, F:¥ADPCM_LIB¥ORCH,　F:¥ADPCM_LIB¥RAP,
F:¥ADPCM_LIB¥ADDITION

を検索するようになる。補助環境変数を用いた場合でも、環境変数'zmusic_PCM'は通常の検索パス表記との混在は可能。

例

SET zmusic_PCM=/0 F:¥ADPCM_LIB;A:¥ZPD_DATA;A:¥PCMFILES;F:¥BOSPCM;

SET zmusic_PCM0=BASS;SNARE;TOMTOM;CYMBAL;ETHNIC;EFFECTS;ACCENT;ORCH;RAP;ADDITION

この例では、

F:¥ADPCM_LIB¥BASS,　F:¥ADPCM_LIB¥SNARE, F:¥ADPCM_LIB¥TOMTOM,
F:¥ADPCM_LIB¥CYMBAL, F:¥ADPCM_LIB¥ETHNIC,F:¥ADPCM_LIB¥EFFECTS,
F:¥ADPCM_LIB¥ACCENT, F:¥ADPCM_LIB¥ORCH,　F:¥ADPCM_LIB¥RAP,
F:¥ADPCM_LIB¥ADDITION

を検索したあと、A:¥ZPD_DATA　A:¥PCMFILES　F:¥BOSPCMを検索する。

なお、Z-MUSIC ver.2.0から利用することができた環境変数'zmusic'の検索プロセスはそのまま利用できるが、優先順位は環境変数'zmusic_???'よりも低く設定されている。

MEASURE 3

X-BASIC用外部関数MUSICZ3.FNC

X680x0本体付属のBASIC「X-BASIC」からZ-MUSIC ver.3.0を使う
方法について述べます。

3．1．　はじめに

X-BASICからZ-MUSICを使うには外部関数「MUSICZ3.FNC」を組み込んだX-BASICを起動する必要がある。

組み込み方は以下のようにして行う。

1)　BASICディレクトリ中のBASIC.CNFの内容のうち、

　　FUNC＝MUSIC

の部分を、

　　FUNC＝MUSICZ3

のように変更する。

2)　このディレクトリにMUSICZ3.FNCをコピーする。

3)　コマンドプロンプトから、

　　A>basic

としてX-BASICを起動する。

しかし、各ユーザーのシステム環境によっては多少の食い違いもあるので、よくわからない人はX680x0付属の「BASICマニュアル」を参照してほしい。

3.2. MUSICZ.FNC(ver.2.0用)との違い

Z-MUSIC ver.2.0用のX-BASIC用外部関数「MUSICZ.FNC」の命令はZ-MUSIC ver.3.0用のX-BASIC外部関数「MUSICZ3.FNC」上で実行することはできるがX-BASICプログラムに記述した場合のみに限られる。コマンドをX-BASICコマンドラインに直接打ち込んでの実行はできない。

また、MUSICZ.FNCの命令を使ったX-BASICプログラムは音楽演奏を行うミュージックプログラムに限られ、コマンド(関数)の戻り値を利用して実行制御を行っているようなツールの動作は保証されない。また、XCでコンパイルも行えない。

逆にMUSICZ3.FNCの命令はMUSICZ.FNCのものと異なり、X-BASICコマンドラインに直接打ち込んでの実行も行え、コンパイルも可能。コマンド(関数)の戻り値を利用して実行制御するようなツールを制作することもできる。

MUSICZ3.FNCにおけるMUSICZ.FNCの命令実行機能は、互換性維持のためだけに設けてあるため、曲の制作はZMSベースで行うことを奨励する。MUSICZ3.FNCはX-BASIC上でミュージックツールなどを制作する場合などに利用するとよい。

MUSICZ3.FNCでは仕様変更のなされたMUSICZ.FNC命令

MUSICZ.FNCとMUSICZ3.FNCとでは以下の関数においてパラメータの与え方や仕様に関して変更が加えられた。

- m_alloc()はダミー命令となった。
- m_stat()はパラメータ省略不可となり、さらにパラメータはチャンネルからトラック番号に変更された。
- m_rec()はI/F番号をパラメータとして取るようになった。(ほぼ同等の機能のzm_midi_rec()を新設)
- m_rstop()はI/F番号をパラメータとして取るようになった。(ほぼ同等の機能のzm_midi_rec_end()を新設)
- m_save()はダミー命令となった。
- m_total()はダミー命令となった。(ほぼ同機能のzm_calc_total()を新設)
- zm_work()のパラメータであるトラック番号の有効範囲は1～80から0～65534に変更にされた。

3.3. X-BASIC音楽プログラムをZMSへ変換する

Z-MUSIC ver.2.0用のX-BASIC用外部関数「MUSICZ.FNC」の命令を利用したX-BASIC音楽プログラムを次の方法でZMSに変換することができる。

1)　プログラムの先頭、あるいはX-BASICコマンドプロンプトから、

　　zm_switch(1,"TEMP.ZMS")

を入力する。1は変換出力スイッチの設定で、'TEMP.ZMS'は出力ファイルのファイルネームに相当する。

2)　変換したいX-BASIC音楽プログラムをRUNする。

以上の手順でZMSが出力される。

変換できるのはZ-MUSIC ver.2.0用のX-BASIC用外部関数「MUSICZ.FNC」の命令のみでZ-MUSIC ver.3.0用のX-BASIC外部関数「MUSICZ3.FNC」の命令は変換できない。

3.4.　コマンドの使用にあたっての注意

MIDI楽器専用のコマンドは場合によっては楽器のメモリを書き換える。十分に楽器側のマニュアルを読んでから実行すること。楽器側に大切なデータが存在する場合は、楽器のメモリをファイルに保存しておくとよい(MEASURE9参照)。

3.5.　MUSICZ3.FNC命令一覧

[注意]

備考欄の、

　　se本命令tup(7):Output Assign　　(0～4:Dry,Rev,Cho,Dir1,Dir2)

といった表記は実際にsetup(7)に格納すべき値が0～4で、これに対応する音源上でのパラメータがDry,Rev,Cho,Dir1,Dir2ということを表している。

つまりこの例でいけば、0がDryに、1がRev、2がChoに対応するということ。

zm_detect(mode)

機能　MUSICZ.FNC(ver.2.0)命令実行におけるエラーの発生場所検出精度設定

引数　int mode:0=通常モード 1=高精度モード

戻り値　なし

備考

ver.2.0のMUSICZ.FNCの命令でエラーが発生してもm_play()の部分で、

　「コンパイラでエラーを発生した」

としか報告されない。

本命令で高精度設定にすると行単位でエラーを検出できるようになるがコンパイル時間が著しく長くなる(MUSICZ.FNC(ver.2.0)ときと比べて4～8倍。ver.2.0以前とver.3.0以降ではコンパイル方式がまったく違うため、そのエミュレーションに手間がかかってしまう)。

MUSICZ3.FNC(ver.3.0)の命令のエラー検出精度はこの命令による設定とは無関係に、常に、この命令でいう高精度モードで行われ、行単位でエラーが検出される。

zm_switch(sw, fn)

機能　ZMSファイルの変換出力スイッチ

引数　char sw:0=出力OFF, 1=出力ON

str fn:出力ファイル名

戻り値　なし

備考

fnを省略するとデフォルトファイルネームとして'ZMUSIC.ZMS'がカレントに作成される。

zm_check_zmsc()

機能　Z-MUSICの常駐状態を検査する

引数　なし

戻り値　-1:エラー(常駐していなかった)
$3x4x:バージョンID

備考

バージョンIDはたとえばZMSC3.X version 3.12が常駐していたならば$3142が返ってくる。4の部分は固定値。

zm_init(param)

機能　Z-MUSICと音源の初期化を行う

引数　int param:0(dummy)

戻り値　Z-MUSICのバージョンID

備考

paramは将来拡張用で必ず0を設定。

バージョンIDは、たとえばバージョン3.21ならば $00003241となる。4の部分は固定値。

zm_assign(device, track)

機能　チャンネルとトラックの割り付けを設定する

引数
int device(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
-1=カレントMIDI
int device(下位16ビット):チャンネル番号0～15
int track:トラック番号0～65534

戻り値　0:正常終了
0以外:エラーコード

備考

カレントMIDIはMIDIインタフェイス番号を指定しない場合のデフォルトとなるものでzm_current_midi_out()で設定する。

MUSICZ3.FNCでは、トラック番号は1～65535ではなく、内部表現の0～65534で表される点に注意。これ以降のコマンドについても同様。

zm_vget(timbre, mode, buffer)

機能　FM音源音色を取り出す

引数
int timbre:0～32767
int mode:0=OPMDRV.X互換,1=AL/FB分離方式
dim char buffer(4,10):音色データ格納配列

戻り値　0:正常終了
0以外:エラーコード

備考

dim char buffer(4,10)で表される音色データの書式(m,n)は以下の通り。

●音色フォーマット(OPMDRV.X互換方式)

(0,0):AF(ﾌｨｰﾄﾞﾊﾞｯｸ/ｱﾙｺﾞﾘｽﾞﾑ) (0～63)
(0,1):OM(ｽﾛｯﾄﾏｽｸ) (0～15)
(0,2):WF(ｳｪｰﾌﾞﾌｫｰﾑ) (0～3)
(0,3):SYC(ｼﾝｸﾛ) (0,1)
(0,4):SPD(ｽﾋﾟｰﾄﾞ) (0～255)
(0,5):PMD (0～127)
(0,6):AMD (0～127)
(0,7):PMS (0～7)
(0,8):AMS (0～3)
(0,9):PAN (0～3)
(0,10):DUMMY
(1,0):OP1, (2,0):OP2, (3,0):OP3, (4,0):OP4
AR(ｱﾀｯｸﾚｰﾄ) (0～31)
(1,1):OP1, (2,1):OP2, (3,1):OP3, (4,1):OP4
1DR(1stﾃﾞｨｹｲﾚｰﾄ) (0～31)
(1,2):OP1, (2,2):OP2, (3,2):OP3, (4,2):OP4
2DR(2ndﾃﾞｨｹｲﾚｰﾄ) (0～31)
(1,3):OP1, (2,3):OP2, (3,3):OP3, (4,3):OP4
RR(ﾘﾘｰｽﾚｰﾄ) (0～15)
(1,4):OP1, (2,4):OP2, (3,4):OP3, (4,4):OP4
1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ) (0～15)
(1,5):OP1, (2,5):OP2, (3,5):OP3, (4,5):OP4
TL(ﾄｰﾀﾙﾚﾍﾞﾙ) (0～127)
(1,6):OP1, (2,6):OP2, (3,6):OP3, (4,6):OP4
KS(ｷｰｽｹｰﾘﾝｸﾞ) (0～3)
(1,7):OP1, (2,7):OP2, (3,7):OP3, (4,7):OP4
MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ) (0～15)
(1,8):OP1, (2,8):OP2, (3,8):OP3, (4,8):OP4
DT1(ﾃﾞｨﾁｭｰﾝ1) (0～7)
(1,9):OP1, (2,9):OP2, (3,9):OP3, (4,9):OP4
DT2(ﾃﾞｨﾁｭｰﾝ2) (0～3)
(1,10):OP1,(2,10):OP2,(3,10):OP3,(4,10):OP4
AME(AMSｲﾈｰﾌﾞﾙ) (0,1)

●音色フォーマット(AL/FB分離方式)

(0,0):OP1, (1,0):OP2, (2,0):OP3, (3,0):OP4
AR(ｱﾀｯｸﾚｰﾄ) (0～31)
(0,1):OP1, (1,1):OP2, (2,1):OP3, (3,1):OP4
1DR(1stﾃﾞｨｹｲﾚｰﾄ) (0～31)
(0,2):OP1, (1,2):OP2, (2,2):OP3, (3,2):OP4
2DR(2ndﾃﾞｨｹｲﾚｰﾄ) (0～31)
(0,3):OP1, (1,3):OP2, (2,3):OP3, (3,3):OP4
RR(ﾘﾘｰｽﾚｰﾄ) (0～15)
(0,4):OP1, (1,4):OP2, (2,4):OP3, (3,4):OP4
1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ) (0～15)
(0,5):OP1, (1,5):OP2, (2,5):OP3, (3,5):OP4
TL(ﾄｰﾀﾙﾚﾍﾞﾙ) (0～127)
(0,6):OP1, (1,6):OP2, (2,6):OP3, (3,6):OP4
KS(ｷｰｽｹｰﾘﾝｸﾞ) (0～3)
(0,7):OP1, (1,7):OP2, (2,7):OP3, (3,7):OP4
MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ) (0～15)
(0,8):OP1, (1,8):OP2, (2,8):OP3, (3,8):OP4
DT1(ﾃﾞｨﾁｭｰﾝ1) (0～7)
(0,9):OP1, (1,9):OP2, (2,9):OP3, (3,9):OP4
DT2(ﾃﾞｨﾁｭｰﾝ2) (0～3)
(0,10):OP1,(1,10):OP2,(2,10):OP3,(3,10):OP4
AME(AMSｲﾈｰﾌﾞﾙ) (0,1)
(4,0):AL(ｱﾙｺﾞﾘｽﾞﾑ) (0～7)
(4,1):FB(ﾌｨｰﾄﾞﾊﾞｯｸ) (0～7)
(4,2):OM(ｽﾛｯﾄﾏｽｸ) (0～15)
(4,3):PAN (0～3)
(4,4):WF(ｳｪｰﾌﾞﾌｫｰﾑ) (0～3)
(4,5):SYC(ｼﾝｸﾛ) (0,1)
(4,6):SPD(ｽﾋﾟｰﾄﾞ) (0～255)
(4,7):PMD (0～127)
(4,8):AMD (0～127)
(4,9):PMS (0～7)
(4,10):AMS (0～3)

zm_vset(timbre, mode, buffer)

機能　FM音源音色を登録する

引数
int timbre:0～32767
int mode:0=OPMDRV.X互換,1=AL/FB分離方式
dim char buffer(4,10):音色データ格納配列

戻り値　0:正常終了
0以外:エラーコード

備考

音色フォーマットについてはzm_vget()とまったく同じ

zm_tempo(tempo, mode)

機能　テンポを設定する

引数
int tempo:テンポの値(1分間の4分音符の数),
-1=テンポ取得のみを行う
int mode:0=Z-MUSICに登録する(通常)
-1=単なるテンポの変更でZ-MUSICには登録しない

戻り値　設定前のテンポとタイマ値が戻ってくる
上位16ビット:タイマ値
下位16ビット:テンポ値

備考

テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。

TIMER-A:20～32767
TIMER-B:20～5000
YM3802:1～32767

zm_set_timer_value(timer, mode)

機能　テンポをタイマ値で設定する
引数　int timer:タイマの値,-1タイマ値取得のみを行う
　　　int mode:0=Z-MUSICに登録する(通常)
　　　　　　-1=単なるテンポの変更でZ-MUSICには登録しない
戻り値　設定前のテンポとタイマ値が戻ってくる
　　　上位16ビット:テンポ値
　　　下位16ビット:タイマ値
備考
　テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。
　　　TIMER-A:0～1023
　　　TIMER-B:0～255
　　　YM3802:0～8191

zm_play(track)

機能　演奏開始制御
引数　dim int(char) track(～):演奏を開始したいトラック番号の並び
戻り値　なし
備考
　dim int track()は、
　　{ トラック番号*65536+トラック番号, トラック番号*65536+トラック番号,…-1}
という構成を取る。ただしトラック番号は0～65534の範囲。

zm_play_all()

機能　全トラック演奏開始制御
引数　なし
戻り値　なし

zm_play2()

機能　演奏開始制御
引数　なし
戻り値　なし
備考
　現在Z-MUSICの演奏データバッファに存在する演奏データを初めから(共通コマンドの実行も含めて)演奏する。

zm_play_again()

機能　演奏開始制御
引数　なし
戻り値　なし
備考
　前回行われたzm_play()を再び実行する

zm_play_status_all_ch(channel_list)

機能　全チャンネル演奏状態検査
引数　dim int(char) channel_list(～):
　　　　　　　　　検査結果を格納するための配列
戻り値　なし
備考
　演奏状態であるチャンネルIDがdim int channel_list()に格納される。
　　{ チャンネルID,チャンネルID,……,-1 }
のような構成を取る。
　バッファサイズは演奏している(トラック数+1)*4バイト必要。
　なおチャンネルIDは、
　　(上位16ビット):0=FM音源,1=AD PCM音源
　　　　　$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
　　　　　-1=カレントMIDI
　　(下位16ビット):チャンネル番号0～15
で表される。

zm_play_status_all_tr(track_list)

機能　全トラック演奏状態検査
引数　dim int(char) track_list(～):検査結果を格納するための配列
戻り値　なし
備考
　演奏状態であるトラック番号(0～65534の範囲)がdim int track_list()に格納される。
　　{ トラック番号*65536+トラック番号,
　　　　　　トラック番号*65536+トラック番号,…,-1 }
のような構成を取る。
　配列の要素数は演奏している(トラック数+1)必要。

zm_play_status_ch(channel)

機能　チャンネル演奏状態検査
引数　int channel(上位16ビット):0=FM音源,1=AD PCM音源
　　　　　$8000=MIDI1,$8001=MIDI2,
　　　　　$8002=MIDI3,-1=カレントMIDI
　　　int channel(下位16ビット):チャンネル番号0～15
戻り値　0:演奏されていない
　　　0以外:演奏されている

zm_play_status_tr(track)

機能　トラック演奏状態検査
引数　int track:検査対象トラック番号(0～65534)
戻り値　0:演奏されていない
　　　0以外:演奏されている

zm_stop(track)

機能　演奏停止制御
引数　dim int(char) track(～):演奏を停止したいトラック番号の並び
戻り値　なし
備考
　dim int track()は、
　　{ トラック番号*65536+トラック番号,
　　　トラック番号*65536+トラック番号,…-1}
という構成を取る。ただしトラック番号は0～65534の範囲

zm_stop_all()

機能　全トラック演奏停止制御
引数　なし
戻り値　なし

zm_cont(track)

機能　演奏再開制御
引数　dim int(char) track(～):演奏を再開したいトラック番号の並び
戻り値　なし
備考
　dim int track()は、
　　{ トラック番号*65536+トラック番号,
　　　トラック番号*65536+トラック番号,…-1}
という構成を取る。ただしトラック番号は0～65534の範囲

zm_cont_all()

機能　全トラック演奏再開制御
引数　なし
戻り値　なし

zm_atoi(track)

機能　演奏トラックデータの先頭アドレス取得
引数　int track:トラック番号0～65534
戻り値　0:エラー
　　　0以外:指定演奏トラックの先頭アドレス

zm_set_master_clock(mstrclk)

機能　拍子,メトロノーム速度,調号,全音符の絶対音長の設定
引数　int side:0=音楽演奏,1=効果音
　　　dim char(int) mstrclk(7):
　　　拍子,メトロノーム速度,調号,全音符の絶対音長を格納した配列
戻り値　なし
備考
　side=0で通常音楽演奏側の設定、side=1で効果音演奏側の設定となる。
　　　dim char mstrclkは、
　　　dim char mstrclk(7)={
　　　　　meter_n,/*拍子分子(1～255)*/
　　　　　meter_d,/*拍子分母(1～255)*/

```
        metronome,      /*音楽的音長(1～255)*/
        dummy,
        key1,   /*調号1(0:C,負値:♭の数,正値:#の数)*/
        key2,           /*調号2(Major=0,Minor=1)*/
        master_clock_H, /*全音符の絶対音長(1～65535)
                                          の上位16ビット*/
        master_clock_L  /*全音符の絶対音長(1～65535)
                                          の下位16ビット*/
    };
```

zm_play_zmd(size, zmd)

機能　ZMDの演奏

引数　int size:ZMDのサイズ

(ヘッダ$1A,'ZmuSiC',$30の分は含まれていても含まれていなくてもどちらでもよい)。

0を設定するとZ-MUSIC本体のバッファへ演奏データを取り込む動作を省略する。

dim char(int) zmd(～):ZMDを格納している配列(ヘッダを含まず)

戻り値　0:正常終了

0以外:エラーコード

備考

int size=0とするには、dim char zmd()からの領域の存在が演奏中も保証されることを前提とする。

zm_play_zmd_se(zmd)

機能　ZMDを効果音として演奏する

引数　dim char(int) zmd(～):ZMDの格納配列(ヘッダを含まず)

戻り値　0:正常終了

0以外:エラーコード

備考

dim char zmd()からの領域の存在は演奏中も保証されなければならない。

zm_se_play(zmd)

機能　効果音ZMDの演奏

引数　dim char(int) zmd(～):ZMDの格納配列

(トラックチャンネル情報テーブルから)

戻り値　0:正常終了

0以外:エラーコード

備考

dim char zmd()からの領域の存在は演奏中も保証されなければならない。

トラックチャンネル情報テーブルについてはMEASURE12を参照。

zm_se_adpcm1(data_type,volume,frq,pan,size,priority,ch,adpcm)

機能　(AD)PCM効果音の演奏

引数　char data_type:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)

char volume:音量(0～127,64:標準)

char frq:再生周波数

(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz)

char pan:パンポット(0～3 / $80+0～127)

int size:再生するAD PCMのデータサイズ

int priority:効果音優先度(0～255/0がもっとも低い)

int ch:再生チャンネル(0～15)

dim char(int) adpcm(～):(AD)PCMデータを格納している配列

戻り値　0:正常終了

0以外:エラーコード

備考

dim char adpcm()からの領域の存在は、演奏中も保証されなければならない。

zm_se_adpcm2(data_type,volume,frq,pan,data_number,priority,ch)

機能　Z-MUSICに登録してある(AD)PCM効果音の演奏

引数　char data_type:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)

char volume:音量(0～127,64:標準)

char frq:パンポット(0～3 / $80+0～127)

char pan:再生周波数

(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz)

int data_number:PCMデータ番号

(d15が1ならば音色番号($8000+0～32767),

0ならばノート番号(0～32767)指定)

int priority:効果音優先度(0～255/0がもっとも低い)

int ch:再生チャンネル(0～15)

戻り値　0:正常終了

0以外:エラーコード

zm_intercept_play(mode)

機能　演奏開始制御の遮断制御

引数　int mode:-1 演奏開始の遮断を行う

0　演奏開始の遮断の解除と演奏開始

1　演奏開始の遮断の解除

戻り値　なし

備考

演奏開始の遮断を設定するとzm_play()などの演奏開始命令を実行しても演奏が開始されなくなる。

zm_current_midi_in(port)

機能　カレントMIDI-IN端子の設定

引数　int port:MIDIポートの番号(0～3)

-1では問い合わせのみを行う

戻り値　以前のカレントMIDI-INのポート番号(0～2)

zm_current_midi_out(port)

機能　カレントMIDI-OUT端子の設定

引数　int port:MIDIポートの番号(0～3)

-1では問い合わせのみを行う

戻り値　以前のカレントMIDI-OUTのポート番号(0～2)

zm_midi_transmission(port,size,midi)

機能　MIDIデータメッセージの送信

引数　int port:MIDIポートの番号(0～3)

-1はカレントMIDI-OUTを使用

int size:送信データサイズ

0とした場合はテキスト型MIDIデータ(MDD)と見なして送信する。0以外ではバイナリデータと見なして送信する

dim char(int) midi(～):送信するMIDIデータを格納した配列

戻り値　送信データの8ビット加算合計値

備考

MDDとはZ-MUSIC v2.0相当でMIDIファイルの標準としていたフォーマット。$0d,$0aで始まり16進数のテキストでMIDIメッセージを記述したファイル。送信データサイズを指定した場合はバイナリのMIDIメッセージと見なして送信される。Z-MUSICが扱えるバイナリのMIDIメッセージにはスタンダードMIDIファイル(FORMAT0)と単なるバイナリデータ列の2種類がありこのデータの種類の判別はZ-MUSICが自動で行う。

zm_exclusive(port,size,id,midi)

機能　エクスクルーシブメッセージの送信

引数　int port:MIDIポートの番号(0～3)

-1はカレントMIDI-OUTを使用

int size:送信データサイズ

int id:d0-d7=MODEL ID(0～127)

d8-d15=DEVICE ID(0～127)

d16-d23=MAKER ID(0～127)

dim char(int) midi(～):送信するMIDIデータを格納した配列

戻り値　なし

備考

エクスクルーシブヘッダの$F0とEOXの$F7は自動送信される。idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

MAKER IDをRoland($41)とした場合はデータ列をローランドエクスクルーシブフォーマットのデータと見なし、コマンドメッセージ(DT2:$12)やチェックサムバイト付きで送信する。

MAKER IDをYAMAHA($43)とした場合はデータ列をヤマハエクスクルーシブフォーマットのデータと見なし、データ長やチェックサムバイト付きで送信する。

zm_set_eox_wait(port,wait)

機能　EOX送信後のウェイトを設定する

引数　int port:MIDIポートの番号(0～3)

-1はカレントMIDI-OUTを使用

int wait:0〜65535
戻り値　それまでのウェイト値

zm_midi_inp1(port,mode)

機能　MIDIデータの1バイト入力
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int mode:0 1回受信を試みてすぐ帰還
　　　　1 受信するまで無限ループ
戻り値　d31=1　読みこぼし発生　0:正常読み出し
　　d15=1　データなし　0:データあり
　　d0-d7　読み出しデータ

zm_midi_out1(port,data)

機能　MIDIデータの1バイト出力
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　char data:送信データ(0〜255)
戻り値　0:正常終了
　　0以外:エラーコード

zm_midi_rec(port)

機能　MIDIデータのレコーディング開始
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
戻り値　0:正常終了
　　0以外:エラーコード

zm_midi_rec_end(port)

機能　MIDIデータのレコーディング終了
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
戻り値　0:正常終了
　　0以外:エラーコード

zm_gs_reset(port, id)

機能　GS音源の初期化
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int id:DEVICE ID(0〜127,-1:current)
戻り値　0:正常終了
　　0以外:エラーコード

zm_gs_partial_reserve(port,id,rsv)

機能　GS音源のパーシャルリザーブ
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int id:DEVICE ID(0〜127,-1:current)
　　dim char(int) rsv(15):パート1〜パート16までの
　　　　パーシャルリザーブ値が格納されている配列
戻り値　0:正常終了
　　0以外:エラーコード
備考
　dim char rsv()にパーシャルリザーブ値を16パート分格納しておく。また、合計が楽器の最大同時発声数以下にならなければならない。
　　rsv(0)=パート1,rsv(1)=パート2…,rsv(15)=パート16
というふうに対応する。なお、パート10はリズムパートに対応する。
　DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_gs_reverb(port,size,id,rvb)

機能　GS音源のリバーブパラメータの設定
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int size:リバーブパラメータデータサイズ(1〜7)
　　int id:DEVICE ID(0〜127,-1:current)
　　dim char(int) rvb(0〜6):リバーブパラメータ格納配列
戻り値　0:正常終了
　　0以外:エラーコード
備考
　dim char rvb()にリバーブパラメータを記述しておく。sizeは7以内で、指定したsize個分のパラメータがGS音源へ送信される。
　DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される

rvb(0):REVERB MACRO 00:Room1,01:Room2,02:Room3,03:Hall1,
　　04:Hall2,05:Plate,06:Delay,07:Panning Delay(FB)
rvb(1):REVERB CHARACTER　($00〜$07)
rvb(2):REVERB PRE-LPF　($00〜$07)
rvb(3):REVERB LEVEL　($00〜$7F)
rvb(4):REVERB TIME　($00〜$7F)
rvb(5):REVERB DELAY FEEDBACK　($00〜$7F)
rvb(6):REVERB SEND LEVEL TO CHORUS　($00〜$7F)
(SC-55マニュアルp79参照)

zm_gs_chorus(port,size,id,cho)

機能　GS音源のコーラスパラメータの設定
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int size:コーラスパラメータデータサイズ(1〜8)
　　int id:DEVICE ID(0〜127,-1:current)
　　dim char(int) cho(0〜7):コーラスパラメータ格納配列
戻り値　0:正常終了
　　0以外:エラーコード
備考
　dim char cho()からの領域にコーラスパラメータを記述しておく。
　sizeは8以内で指定したsize個分のパラメータがGS音源へ送信される。
　DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

cho(0):CHORUS MACRO 00:Chorus1,01:Chorus2,02:Chorus3,
　　03:Chorus4,04:Feedback Chorus,05:Flanger,
　　06:Short Delay,07:Short Delay(FB)
cho(1):CHORUS PRE-LPF　($00〜$07)
cho(2):CHORUS LEVEL　($00〜$7F)
cho(3):CHORUS FEEDBACK　($00〜$7F)
cho(4):CHORUS DELAY　($00〜$7F)
cho(5):CHORUS RATE　($00〜$7F)
cho(6):CHORUS DEPTH　($00〜$7F)
cho(7):CHORUS SEND LEVEL TO REVERB　($00〜$7F)
(SC-55マニュアルp79参照)

zm_gs_part_parameter(port,size,pt_id,param)
zm_gs_part_setup(port,size,pt_id,param)

機能　GS音源のパートパラメータの設定
引数　int port:MIDIポートの番号(0〜3)
　　　　-1はカレントMIDI-OUTを使用
　　int size:パートパラメータデータサイズ(1〜119)
　　int pt_id:d0-d15=DEVICE ID(0〜127,-1:current)
　　　　d16-d31=PART NUMBER(1〜16)
　　dim char(int) param(0〜118):パートパラメータ格納配列
戻り値　0:正常終了
　　0以外:エラーコード
備考
　dim char param()にパートパラメータを記述しておく。sizeは119以内で指定したsize個分のパラメータがGS音源へ送信される。
　DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

param(0):Rx CHANNEL　(1〜16,17=OFF)
param(1):Rx PITCH BEND　(0〜1:OFF/ON)
param(2):Rx CH PRESSURE(CAf)　(0〜1:OFF/ON)
param(3):Rx PROGRAM CHANGE　(0〜1:OFF/ON)
param(4):Rx CONTROL CHANGE　(0〜1:OFF/ON)
param(5):Rx POLY PRESSURE(PAf)　(0〜1:OFF/ON)
param(6):Rx NOTE MESSAGE　(0〜1:OFF/ON)
param(7):Rx RPN　(0〜1:OFF/ON)
param(8):Rx NRPN　(0〜1:OFF/ON)
param(9):Rx MODURATION　(0〜1:OFF/ON)
param(10):Rx VOLUME　(0〜1:OFF/ON)
param(11):Rx PANPOT　(0〜1:OFF/ON)
param(12):Rx EXPRESSION　(0〜1:OFF/ON)
param(13):Rx HOLD1　(0〜1:OFF/ON)

param(14):Rx PORTAMENTO (0～1:OFF/ON)
param(15):Rx SOSTENUTO (0～1:OFF/ON)
param(16):Rx SOFT (0～1:OFF/ON)
param(17):MONO/POLY MODE (0～1:MONO/POLY)
param(18):ASSIGN MODE
(0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
param(19):USE FOR RHYTHM PART (0:OFF,1:MAP1,2:MAP2)
param(20):PITCH KEY SHIFT
($28～$58:－24～＋24[semitone])
param(21):PITCH OFFSET FINE(upper)
($08～$F8:－12.0～＋12.0[Hz])
param(22):PITCH OFFSET FINE(lower) 〃
param(23):PART LEVEL (0～127)
param(24):VELOCITY SENSE DEPTH (0～127)
param(25):VELOCITY SENSE OFFSET (0～127)
param(26):PART PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
param(27):KEY RANGE LOW (0～127:C-1～G9)
param(28):KEY RANGE HIGH (0～127:C-1～G9)
param(29):CC1 CONTROL NUMBER (0～127)
param(30):CC2 CONTROL NUMBER (0～127)
param(31):CHORUS SEND DEPTH (0～127)
param(32):REVERB SEND DEPTH (0～127)

param(33):Vibrato rate ($0E～$72:－50～＋50)
param(34):Vibrato depth ($0E～$72:－50～＋50)
param(35):TVF cutoff freq. ($0E～$50:－50～＋16)
param(36):TVF resonance ($0E～$72:－50～＋50)
param(37):TVF&TVA Env.attack ($0E～$72:－50～＋50)
param(38):TVF&TVA Env.decay ($0E～$72:－50～＋50)
param(39):TVF&TVA Env.release ($0E～$72:－50～＋50)
param(40):Vibrato delay ($0E～$72:－50～＋50)

param(41):SCALE TUNING C ($00～$7F:－64～＋63[cent])
param(42):SCALE TUNING C# ($00～$7F:－64～＋63[cent])
param(43):SCALE TUNING D ($00～$7F:－64～＋63[cent])
param(44):SCALE TUNING D# ($00～$7F:－64～＋63[cent])
param(45):SCALE TUNING E ($00～$7F:－64～＋63[cent])
param(46):SCALE TUNING F ($00～$7F:－64～＋63[cent])
param(47):SCALE TUNING F# ($00～$7F:－64～＋63[cent])
param(48):SCALE TUNING G ($00～$7F:－64～＋63[cent])
param(49):SCALE TUNING G# ($00～$7F:－64～＋63[cent])
param(50):SCALE TUNING A ($00～$7F:－64～＋63[cent])
param(51):SCALE TUNING A# ($00～$7F:－64～＋63[cent])
param(52):SCALE TUNING B ($00～$7F:－64～＋63[cent])

param(53):MOD PITCH CONTROL ($28～$58:－24～＋24[semitone])
param(54):MOD TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(55):MOD AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(56):MOD LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(57):MOD LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(58):MOD LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(59):MOD LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(60):MOD LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(61):MOD LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param(62):MOD LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param(63):MOD LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param(64):BEND PITCH CONTROL
($28～$58:－24～＋24[semitone])
param(65):BEND TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(66):BEND AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(67):BEND LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(68):BEND LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(69):BEND LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(70):BEND LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(71):BEND LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(72):BEND LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param(73):BEND LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param(74):BEND LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param(75):CAf PITCH CONTROL
($28～$58:－24～＋24[semitone])
param(76):CAf TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(77):CAf AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(78):CAf LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(79):CAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(80):CAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(81):CAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(82):CAf LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(83):CAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param(84):CAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param(85):CAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param(86):PAf PITCH CONTROL
($28～$58:－24～＋24[semitone])
param(87):PAf TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(88):PAf AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(89):PAf LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(90):PAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(91):PAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(92):PAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(93):PAf LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(94):PAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param(95):PAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param(96):PAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param(97):CC1 PITCH CONTROL ($28～$58:－24～＋24[semitone])
param(98):CC1 TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(99):CC1 AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(100):CC1 LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(101):CC1 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(102):CC1 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(103):CC1 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(104):CC1 LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(105):CC1 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param(106):CC1 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param(107):CC1 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param(108):CC2 PITCH CONTROL($28～$58:－24～＋24[semitone])
param(109):CC2 TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param(110):CC2 AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param(111):CC2 LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param(112):CC2 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param(113):CC2 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param(114):CC2 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param(115):CC2 LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])

param(116):CC2 LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
param(117):CC2 LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
param(118):CC2 LFO2 TVA DEPTH　($00～$7F:0～100.0[%])
(SC-55のマニュアルp75,p79～p80参照)

zm_gs_drum_setup(port,size,dr_id,param)

機能　GS音源のドラムセットアップ設定

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1～8)
int dr_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d23=NOTE NUMBER(0～127)
d24-d31=MAP NUMBER(0～1)
dim char(int) param(0～7):ドラムセットアップパラメータ格納配列

戻り値　0:正常終了
0以外:エラーコード

備考
dim char param()にドラムセットアップパラメータを記述しておく。
sizeは８以内で指定したsize個分のパラメータがGS音源へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

param(0):PLAY KEY NUMBER　(0～127)
param(1):LEVEL　(0～127)
param(2):ASSIGN GROUP NUMBER　(0:NON,1～127)
param(3):PANPOT　(0:RANDOM, 01(LEFT)～64～127(RIGHT))
param(4):REVERB DEPTH　(0～127:0.0～1.0)
param(5):CHORUS DEPTH　(0～127:0.0～1.0)
param(6):Rx NOTE OFF　(0～1:OFF/ON)
param(7):Rx NOTE ON　(0～1:OFF/ON)
(SC-55マニュアルp75,p82参照)

zm_gs_drum_name(port,size,mp_id,name);

機能　GS音源のドラムセット名設定

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1～8)
int mp_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d31=MAP NUMBER(0,1)
str name:ドラムセット名文字列格納アドレス

戻り値　0:正常終了
0以外:エラーコード

備考
文字列は12文字以内でchar *nameからの領域に文字列を記述しておく。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
(SC-55マニュアルp82参照)

zm_gs_print(port,size,id,mes)

機能　GS音源の画面へのメッセージ出力

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:メッセージ長(1～32)
int id:d0-d15=DEVICE ID(0～127,-1:current)
str mes:文字列

戻り値　0:正常終了
0以外:エラーコード

備考
メッセージは32文字以内でstr mesに文字列を記述しておく。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_gs_display(port,id,pattern)

機能　GS音源の画面へのドットパターン出力

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0～127,-1:current)
dim int(char) pattern(7):ドットパターン格納配列

戻り値　0:正常終了
0以外:エラーコード

備考
dim int pattern()にドットパターンを記述しておく。
フォーマットは、
{ 1段目のドットパターン*65536+2段目のドットパターン,
3段目のドットパターン*65536+3段目のドットパターン,
5段目のドットパターン*65536+4段目のドットパターン,
:
15段目のドットパターン*65536+16段目のドットパターン}
に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

例
```
/*SC-55の画面に"善"を表示する*/
dim int d(7)={ %0001000000010000*65536+%0000100000100000,
               %0111111111111100*65536+%0000000100000000,
               %0011111111111000*65536+%0000000100000000,
               %0111111111111100*65536+%0001000100010000,
               %0000100100100000*65536+%1111111111111110,
               %0000000000000000*65536+%0011111111111000,
               %0010000000001000*65536+%0010000000001000,
               %0011111111111000*65536+%0010000000001000}
gs_display(-1,-1,d)
```

zm_gm_system_on(port)

機能　GM音源のリセット

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用

戻り値　0:正常終了
0以外:エラーコード

zm_sc88_mode_set(port,md_id)

機能　SC-88モード設定

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int md_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d31=SC-88 MODE(0:Single, 1:Double)

戻り値　0:正常終了
0以外:エラーコード

備考
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_sc88_reverb(port,size,id,rvb)

機能　SC-88のリバーブパラメータの設定

引数　int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:リバーブパラメータデータサイズ(1～8)
int id:DEVICE ID(0～127,-1:current)
dim char(int) rvb(7):リバーブパラメータ格納配列

戻り値　0:正常終了
0以外:エラーコード

備考
dim char rvb()にリバーブパラメータを記述しておく。sizeは8以内で、指定したsize個分のパラメータがSC-88へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される

rvb(0):REVERB MACRO　(00:Room1,01:Room2, 02:Room3,03:Hall1, 04:Hall2,05:Plate, 06:Delay, 07:Panning Delay(FB))
rvb(1):REVERB CHARACTER　($00～$07)
rvb(2):REVERB PRE-LPF　($00～$07)
rvb(3):REVERB LEVEL　($00～$7F)
rvb(4):REVERB TIME　($00～$7F)
rvb(5):REVERB DELAY FEEDBACK　($00～$7F)
rvb(6):REVERB SEND LEVEL TO CHORUS ($00～$7F/SC-88ではダミー)
rvb(7):REVERB PRE-DELAY TIME　($00～$7F)
(SC-88VLマニュアルp77参照)

zm_sc88_chorus(port,size,id,cho)

機能　SC-88のコーラスパラメータの設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int size:コーラスパラメータデータサイズ(1～9)
　　　int id:DEVICE ID(0～127,-1:current)
　　　dim char(int) cho(8):コーラスパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char cho()にコーラスパラメータを記述しておく。sizeは9以内で指定したsize個分のパラメータがSC-88へ送信される。

DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
cho(0):CHORUS MACRO            (00:Chorus1,01:Chorus2,
                                 02:Chorus3,03:Chorus4,
                                 04:Feedback Chorus,
                                 05:Flanger,
                                 06:Short Delay,
                                 07:Short Delay(FB))
cho(1):CHORUS PRE-LPF          ($00～$07)
cho(2):CHORUS LEVEL            ($00～$7F)
cho(3):CHORUS FEEDBACK         ($00～$7F)
cho(4):CHORUS DELAY            ($00～$7F)
cho(5):CHORUS RATE             ($00～$7F)
cho(6):CHORUS DEPTH            ($00～$7F)
cho(7):CHORUS SEND LEVEL TO REVERB ($00～$7F)
cho(8):CHORUS SEND LEVEL TO DELAY  ($00～$7F)
(SC-88VLマニュアルp78参照)
```

zm_sc88_delay(port,size,id,dly)

機能　SC-88のディレイパラメータの設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int size:ディレイパラメータデータサイズ(1～11)
　　　int id:DEVICE ID(0～127,-1:current)
　　　dim char dly(0～10):ディレイパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char dly()にリバーブパラメータを記述しておく。sizeは11以内で、指定したsize個分のパラメータがSC-88へ送信される。

DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される

```
dly(0):DELAY MACRO   00:Delay1,01:Delay2,02:Delay3,
  03:Delay4,04:Pan Delay1,05:Pan Delay2,06:Pan Delay3,
  07:Pan Delay4,08:Delay to Reverb,09:Pan Repeat
dly(1):DELAY PRE-LPF               ($00～$07)
dly(2):DELAY TIME CENTER         ($01～$73:0.1ms～sec)
dly(3):DELAY TIME RATIO LEFT       ($01～$78:4～500%)
dly(4):DELAY TIME RATIO RIGHT      ($01～$78:4～500%)
dly(5):DELAY LEVEL CENTER          ($00～$7F)
dly(6):DELAY LEVEL LEFT            ($00～$7F)
dly(7):DELAY LEVEL RIGHT           ($00～$7F)
dly(8):DELAY LEVEL                 ($00～$7F)
dly(9):DELAY FEEDBACK            ($00～$7F:-64～+63)
dly(10):DELAY SENDLEVEL TO REVERB  ($00～$7F)
(SC-88VLマニュアルp78参照)
```

zm_sc88_equalizer(port,size,id,eql)

機能　SC-88のイコライザパラメータの設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int size:イコライザパラメータデータサイズ(1～4)
　　　int id:DEVICE ID(0～127,-1:current)
　　　dim char eql(0～3):イコライザパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char eql()にコーラスパラメータを記述しておく。sizeは4以内で指定したsize個分のパラメータがSC-88へ送信される。

DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
eql(0):EQ LOW FREQ          (0,1:200Hz,400Hz)
eql(1):EQ LOW GAIN          ($34～$4c:-12～+12dB)
eql(2):EQ HIGH FREQ         (0,1:3kHz,6kHz)
eql(3):EQ HIGH GAIN         ($34～$4c:-12～+12dB)
(SC-88VLマニュアルp78参照)
```

zm_sc88_part_parameter(port,size,pt_id,param)

機能　SC-88のパートパラメータの設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int size:パートパラメータデータサイズ(1～127)
　　　int pt_id:d0-d15=DEVICE ID(0～127,-1:current)
　　　　d16-d31=PART NUMBER(1～16)
　　　dim char param(0～126):パートパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char param()にパートパラメータを記述しておく。sizeは127以内で指定したsize個分のパラメータがSC-88へ送信される。

DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
param(0):Rx CHANNEL              (1～16,17=OFF)
param(1):Rx PITCH BEND           (0～1:OFF/ON)
param(2):Rx CH PRESSURE(CAf)     (0～1:OFF/ON)
param(3):Rx PROGRAM CHANGE       (0～1:OFF/ON)
param(4):Rx CONTROL CHANGE       (0～1:OFF/ON)
param(5):Rx POLY PRESSURE(PAf)   (0～1:OFF/ON)
param(6):Rx NOTE MESSAGE         (0～1:OFF/ON)
param(7):Rx RPN                  (0～1:OFF/ON)
param(8):Rx NRPN                 (0～1:OFF/ON)
param(9):Rx MODURATION           (0～1:OFF/ON)
param(10):Rx VOLUME              (0～1:OFF/ON)
param(11):Rx PANPOT              (0～1:OFF/ON)
param(12):Rx EXPRESSION          (0～1:OFF/ON)
param(13):Rx HOLD1               (0～1:OFF/ON)
param(14):Rx PORTAMENTO          (0～1:OFF/ON)
param(15):Rx SOSTENUTO           (0～1:OFF/ON)
param(16):Rx SOFT                (0～1:OFF/ON)
param(17):MONO/POLY MODE         (0～1:MONO/POLY)
param(18):ASSIGN MODE
                    (0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
param(19):USE FOR RHYTHM PART    (0:OFF,1:MAP1,2:MAP2)
param(20):PITCH KEY SHIFT   ($28～$58:－24～＋24[semitone])
param(21):PITCH OFFSET FINE(upper)
                              ($08～$F8:－12.0～＋12.0[Hz])
param(22):PITCH OFFSET FINE(lower)        〃
param(23):PART LEVEL             (0～127)
param(24):VELOCITY SENSE DEPTH   (0～127)
param(25):VELOCITY SENSE OFFSET  (0～127)
param(26):PART PANPOT   (0:RANDOM,01(LEFT)～64～127(RIGHT))
param(27):KEY RANGE LOW          (0～127:C-1～G9)
param(28):KEY RANGE HIGH         (0～127:C-1～G9)
param(29):CC1 CONTROL NUMBER     (0～127)
param(30):CC2 CONTROL NUMBER     (0～127)
param(31):CHORUS SEND LEVEL      (0～127)
param(32):REVERB SEND LEVEL      (0～127)
param(33):Rx BANK SELECT         (0:OFF,1:ON)
param(34):Rx BANK SELECT LSB     (0:OFF,1:ON)

param(35):PITCH FINE TUNE(Upper)
                    ($00 00～$40 00～$7f 7f:-100～0～+100)
param(36):PITCH FINE TUNE(Lower)

param(37):DELAY SEND LEVEL        (0～127)

param(38):Vibrato rate           ($00～$7f:－64～＋63)
param(39):Vibrato depth          ($00～$7f:－64～＋63)
```

param(40):TVF cutoff freq. ($00〜$7f:−64〜+63)
param(41):TVF resonance ($00〜$7f:−64〜+63)
param(42):TVF&TVA Env.attack ($00〜$7f:−64〜+63)
param(43):TVF&TVA Env.decay ($00〜$7f:−64〜+63)
param(44):TVF&TVA Env.release ($00〜$7f:−64〜+63)
param(45):Vibrato delay ($00〜$7f:−64〜+63)

param(46):SCALE TUNING C ($00〜$7F:−64〜+63[cent])
param(47):SCALE TUNING C# ($00〜$7F:−64〜+63[cent])
param(48):SCALE TUNING D ($00〜$7F:−64〜+63[cent])
param(49):SCALE TUNING D# ($00〜$7F:−64〜+63[cent])
param(50):SCALE TUNING E ($00〜$7F:−64〜+63[cent])
param(51):SCALE TUNING F ($00〜$7F:−64〜+63[cent])
param(52):SCALE TUNING F# ($00〜$7F:−64〜+63[cent])
param(53):SCALE TUNING G ($00〜$7F:−64〜+63[cent])
param(54):SCALE TUNING G# ($00〜$7F:−64〜+63[cent])
param(55):SCALE TUNING A ($00〜$7F:−64〜+63[cent])
param(56):SCALE TUNING A# ($00〜$7F:−64〜+63[cent])
param(57):SCALE TUNING B ($00〜$7F:−64〜+63[cent])

param(58):MOD PITCH CONTROL
($28〜$58:−24〜+24[semitone])
param(59):MOD TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(60):MOD AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(61):MOD LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(62):MOD LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(63):MOD LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(64):MOD LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(65):MOD LFO2 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(66):MOD LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(67):MOD LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(68):MOD LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(69):BEND PITCH CONTROL
($40〜$58:−24〜+24[semitone])
param(70):BEND TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(71):BEND AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(72):BEND LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(73):BEND LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(74):BEND LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(75):BEND LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(76):BEND LFO2 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(77):BEND LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(78):BEND LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(79):BEND LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(80):CAf PITCH CONTROL ($28〜$58:−24〜+24[semitone])
param(81):CAf TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(82):CAf AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(83):CAf LFO1 RATE CONTROL($00〜$7F:−10.0〜+10.0[Hz])
param(84):CAf LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(85):CAf LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(86):CAf LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(87):CAf LFO2 RATE CONTROL($00〜$7F:−10.0〜+10.0[Hz])
param(88):CAf LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(89):CAf LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(90):CAf LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(91):PAf PITCH CONTROL ($28〜$58:−24〜+24[semitone])
param(92):PAf TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(93):PAf AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(94):PAf LFO1 RATE CONTROL($00〜$7F:−10.0〜+10.0[Hz])
param(95):PAf LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(96):PAf LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(97):PAf LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(98):PAf LFO2 RATE CONTROL($00〜$7F:−10.0〜+10.0[Hz])
param(99):PAf LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(100):PAf LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(101):PAf LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(102):CC1 PITCH CONTROL($28〜$58:−24〜+24[semitone])
param(103):CC1 TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(104):CC1 AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(105):CC1 LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(106):CC1 LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(107):CC1 LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(108):CC1 LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(109):CC1 LFO2 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(110):CC1 LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(111):CC1 LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(112):CC1 LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(113):CC2 PITCH CONTROL($28〜$58:−24〜+24[semitone])
param(114):CC2 TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param(115):CC2 AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param(116):CC2 LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(117):CC2 LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(118):CC2 LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(119):CC2 LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param(120):CC2 LFO2 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param(121):CC2 LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param(122):CC2 LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param(123):CC2 LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param(124):TONE MAP NUMBER (0:SELECTED,1:SC-55,2:SC-88)
param(125):TONE MAP0 NUMBER (1:SC-55,2:SC-88)
param(126):EQ ON/OFF (0:OFF,1:ON)
(SC-88VLマニュアルp79〜p81参照)

zm_sc88_drum_setup(port,size,dr_id,param)

機能　SC-88のドラムセットアップ設定
引数　int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1〜9)
int dr_id:d0-d15=DEVICE ID(0〜127,-1:current)
d16-d23=NOTE NUMBER(0〜127)
d24-d31=MAP NUMBER(0〜1)
dim char param(0〜8):ドラムセットアップパラメータ格納配列
戻り値　0:正常終了
0以外:エラーコード
備考
dim char param()にドラムセットアップパラメータを記述しておく。
sizeは9以内で指定したsize個分のパラメータがSC-88へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
param(0):PLAY KEY NUMBER (0〜127)
param(1):LEVEL (0〜127)
param(2):ASSIGN GROUP NUMBER (0:NON,1〜127)
param(3):PANPOT (0:RANDOM,01(LEFT)〜64〜127(RIGHT))
param(4):REVERB DEPTH (0〜127:0.0〜1.0)

param(5):CHORUS DEPTH (0～127:0.0～1.0)
param(6):Rx NOTE OFF (0～1:OFF/ON)
param(7):Rx NOTE ON (0～1:OFF/ON)
param(8):DELAY SEND LEVEL (0～127:0.0～1.0)
(SC-88VLマニュアルp82参照)

zm_sc88_drum_name(port,size,mp_id,name);

機能 SC-88のドラムセット名設定

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1～8)
int mp_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d31=MAP NUMBER(0,1/$80,$81)
str name:ドラムセット名文字列格納アドレス

戻り値 0:正常終了
0以外:エラーコード

備考
文字列は12文字以内でchar *nameからの領域に記述しておく。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
MAP NUMBER=$80がユーザードラムセット65,MAP NUMBER=$81がユーザードラムセット66に対応する。
(SC-88VLマニュアルp82参照)

zm_sc88_user_inst(port,size,ui_id,param)

機能 SC-88のユーザー音色の設定

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:ユーザー音色パラメータデータサイズ(1～11)
int ui_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d23=PROGRAM NUMBER(0～127)
d24-d31=BANK NUMBER(0～1)
dim char param(0～10):ユーザー音色パラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char param()にユーザー音色パラメータを記述しておく。
sizeは11以内で指定したsize個分のパラメータがSC-88へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
param(0):SOURCE TONE#(MAP) (1,2)
param(1):BANK NUMBER MSB (0～127)
param(2):PROGRAM NUMBER (0～127)
param(3):Vibrato rate ($00～$7f:－64～＋63)
param(4):Vibrato depth ($00～$7f:－64～＋63)
param(5):TVF cutoff freq. ($00～$7f:－64～＋63)
param(6):TVF resonance ($00～$7f:－64～＋63)
param(7):TVF&TVA Env.attack ($00～$7f:－64～＋63)
param(8):TVF&TVA Env.decay ($00～$7f:－64～＋63)
param(9):TVF&TVA Env.release ($00～$7f:－64～＋63)
param(10):Vibrato delay ($00～$7f:－64～＋63)
(SC-88VLマニュアルp82参照)

zm_sc88_user_drum(port,size,ud_id,param)

機能 SC-88のユーザードラムセットの設定

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:ユーザードラムセットパラメータデータサイズ(1～12)
int ud_id:d0-d15=DEVICE ID(0～127,-1:current)
d16-d23=NOTE NUMBER(0～127)
d24-d31=SET NUMBER(0～1)
dim char param(0～11):
ユーザードラムセットパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
char *paramからの領域にユーザードラムセットパラメータを記述しておく。
sizeは12以内で指定したsize個分のパラメータがSC-88へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
param(0):PLAY KEY NUMBER (0～127)
param(1):LEVEL (0～127)
param(2):ASSIGN GROUP NUMBER (0:NON,1～127)
param(3):PANPOT (0:RANDOM,
01(LEFT)～64～127(RIGHT))
param(4):REVERB DEPTH (0～127:0.0～1.0)
param(5):CHORUS DEPTH (0～127:0.0～1.0)
param(6):Rx NOTE OFF (0～1:OFF/ON)
param(7):Rx NOTE ON (0～1:OFF/ON)
param(8):DELAY SEND LEVEL (0～127:0.0～1.0)
param(9):SOURCE DRUM SET#(MAP) (1～2)
param(10):PROGRAM NUMBER (0～127)
param(11):SOURCE NOTE NUMBER (0～127)
(SC-88VLマニュアルp82参照)

zm_mt32_reset(port,id)

機能 MT-32の初期化

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0～127,-1:current)

戻り値 0:正常終了
0以外:エラーコード

備考
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_mt32_partial_reserve(port,id,rsv)

機能 MT-32のパーシャルリザーブ

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int id:DEVICE ID(0～127,-1:current)
dim char(int) rsv(8):パート1～パート9までのパーシャルリザーブ値が格納されている配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char rsv()にパーシャルリザーブ値を9パート分格納しておく。
また、合計が楽器の最大同時発声数以下にならなければならない。
param(0)＝パート1,param(1)＝パート2…,param(9)＝パート9
というふうに対応する。なお、パート9はリズムパートに対応する。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_mt32_reverb(port,size,id,rsv)

機能 MT-32のリバーブパラメータの設定

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:リバーブパラメータデータサイズ(1～3)
int id:DEVICE ID(0～127,-1:current)
dim char(int) rvb(0～2):リバーブパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char rvb()にリバーブパラメータを記述しておく。sizeは3以内で、指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される
rvb(0):REVERB MODE (0:Room,1:Hall,2:Plate,3:Tap Delay)
rvb(1):REVERB TIME (0～7)
rvb(2):REVERB LEVEL (0～7)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

zm_mt32_part_setup(port,size,id,rsv)

機能 MT-32のパートパラメータの設定

引数 int port:MIDIポートの番号(0～3)
-1はカレントMIDI-OUTを使用
int size:パートパラメータデータサイズ(1～9)
int id:d0-d15=DEVICE ID(0～127,-1:current)
dim char(int) param(0～8):パートパラメータ格納配列

戻り値　0:正常終了
　　　　0以外:エラーコード

備考

dim char param()にパートパラメータを記述しておく。
パートパラメータは、
　param(0)＝パート1のMIDIチャンネル,param(1)＝パート2のMIDIチャンネル…,param(9)＝リズムパートのMIDIチャンネル
という構成を取る

sizeは9以内で指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_mt32_drum(port,size,dr_id,drum)

機能　MT-32のドラムセットアップ設定

引数　int port:MIDIポートの番号(0～3)
　　　　　-1はカレントMIDI-OUTを使用
　　　int size:ドラムセットアップパラメータデータサイズ(1～4)
　　　int dr_id:d0-d15=DEVICE ID(0～127,-1:current)
　　　　　d16-d31=NOTE NUMBER(24～87)
　　　dim char(int) drum(0～3):
　　　　　ドラムセットアップパラメータ格納配列

戻り値　0:正常終了
　　　　0以外:エラーコード

備考

dim char drum()にドラムセットアップパラメータを記述しておく。
sizeは4以内で指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

drum(0):TIMBRE　(0～63:i1～64,64～127:r1～64)
drum(1):OUTPUT LEVEL　(0～100)
drum(2):PANPOT　(0～14)
drum(3):REVERB SWITCH　(0～1:OFF/ON)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

zm_mt32_common(port,size,cm_id,common)

機能　MT-32のコモンパラメータの設定

引数　int port:MIDIポートの番号(0～3)
　　　　　-1はカレントMIDI-OUTを使用
　　　int size:コモンパラメータデータサイズ(1～4)
　　　int cm_id:d0-d15=DEVICE ID(0～127,-1:current)
　　　　　d16-d31=PROGRAM NUMBER(1～64)
　　　dim char(int) common(0～3):コモンパラメータ格納配列

戻り値　0:正常終了
　　　　0以外:エラーコード

備考

dim char common()にコモンパラメータを記述しておく。
sizeは4以内で指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

common(0):Structure of Partial#1&2　(0～12:1～13)
common(1):Structure of Partial#3&4　(0～12:1～13)
common(2):PARTIAL MUTE　(0～15)
common(3):ENV MODE　(0～1:NORMAL,NO SUSTAIN)
(MT-32マニュアルp34,CM-64マニュアルp29参照)

zm_mt32_partial(port,size,pl_id,partial)

機能　MT-32のパーシャルパラメータ設定

引数　int port:MIDIポートの番号(0～3)
　　　　　-1はカレントMIDI-OUTを使用
　　　int size:パーシャルパラメータデータサイズ(1～58)
　　　int pl_id:d0-d15=DEVICE ID(0～127,-1:current)
　　　　　d16-d23=PARTIAL NUMBER(1～4)
　　　　　d24-d31=PROGRAM NUMBER(1～64)
　　　dim char(int) partial(0～57):パーシャルパラメータ格納配列

戻り値　0:正常終了
　　　　0以外:エラーコード

備考

dim char partial()にパーシャルパラメータを記述しておく。
sizeは58以内で指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1 とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

partial(0):WG PITCH COARSE　(0～96:C1,C#1,～,C9)
partial(1):WG PITCH FINE　(0～100:－50～＋50)
partial(2):WG PITCH KEYFOLLOW (0～16:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2,s1,s2)
partial(3):WG PITCH BENDER SW　(0～1:OFF/ON)
partial(4):WG WAVEFORM/PCM BANK　(0～3:SQU1,SAW1,SQU2,SAW2)
partial(5):WG PCM WAVE　(0～127:1～128)
partial(6):WG PULSE WIDTH　(0～100)
partial(7):WG PW VELO SENS　(0～14:－7～＋7)
partial(8):P-ENV DEPTH　(0～10)
partial(9):P-ENV VELO SENS　(0～3)
partial(10):P-ENV TIME KEYF　(0～4)
partial(11):P-ENV TIME 1　(0～100)
partial(12):P-ENV TIME 2　(0～100)
partial(13):P-ENV TIME 3　(0～100)
partial(14):P-ENV TIME 4　(0～100)
partial(15):P-ENV LEVEL 0　(0～100:－50～＋50)
partial(16):P-ENV LEVEL 1　(0～100:－50～＋50))
partial(17):P-ENV LEVEL 2　(0～100:－50～＋50)
partial(18):P-ENV SUSTAIN LEVEL　(0～100:－50～＋50)
partial(19):END LEVEL　(0～100:－50～＋50)
partial(20):P-LFO RATE　(0～100)
partial(21):P-LFO DEPTH　(0～100)
partial(22):P-LFO MOD SENS　(0～100)
partial(23):TVF CUTOFF FREQ　(0～100)
partial(24):TVF RESONANCE　(0～30)
partial(25):TVF KEYFOLLOW (0～14:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2)
partial(26):TVF BIAS POINT/DIR　(0～127)
partial(27):TVF BIAS LEVEL　(0～14:－7～＋7)
partial(28):TVF ENV DEPTH　(0～100)
partial(29):TVF ENV VELO SENS　(0～100)
partial(30):TVF ENV DEPTH KEYF　(0～4)
partial(31):TVF ENV TIME KEYF　(0～4)
partial(32):TVF ENV TIME 1　(0～100)
partial(33):TVF ENV TIME 2　(0～100)
partial(34):TVF ENV TIME 3　(0～100)
partial(35):TVF ENV TIME 4　(0～100)
partial(36):TVF ENV TIME 5　(0～100)
partial(37):TVF ENV LEVEL 1　(0～100)
partial(38):TVF ENV LEVEL 2　(0～100)
partial(39):TVF ENV LEVEL 3　(0～100)
partial(40):TVF ENV SUSTAIN LEVEL　(0～100)
partial(41):TVA LEVEL　(0～100)
partial(42):TVA VELO SENS　(0～100:－50～＋50)
partial(43):TVA BIAS POINT 1　(0～127)
partial(44):TVA BIAS LEVEL 1　(0～12:－12～0)
partial(45):TVA BIAS POINT 2　(0～127)
partial(46):TVA BIAS LEVEL 2　(0～12:－12～0)
partial(47):TVA ENV TIME KEYF　(0～4)
partial(48):TVA ENV TIME V_FOLLOW　(0～4)
partial(49):TVA ENV TIME 1　(0～100)
partial(50):TVA ENV TIME 2　(0～100)
partial(51):TVA ENV TIME 3　(0～100)
partial(52):TVA ENV TIME 4　(0～100)
partial(53):TVA ENV TIME 5　(0～100)
partial(54):TVA ENV LEVEL 1　(0～100)
partial(55):TVA ENV LEVEL 2　(0～100)
partial(56):TVA ENV LEVEL 3　(0～100)
partial(57):TVA ENV SUSTAIN LEVEL　(0～100)
(MT-32マニュアルp34,CM-64マニュアルp29参照)

zm_mt32_patch(port,size,pt_id,patch)

機能　MT-32のパッチパラメータ設定

引数　int port:MIDIポートの番号(0～3)
　　　　　-1はカレントMIDI-OUTを使用
　　　int size:パッチパラメータデータサイズ(1～7)
　　　int pt_id:d0-d15=DEVICE ID(0～127,-1:current)

d16-d31=PATCH NUMBER(1〜128)
dim char(int) patch(0〜6):パッチパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char patch()にパッチパラメータを記述しておく。
sizeは7以内で指定したsize個分のパラメータがMT-32へ送信される。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

patch(0):TIMBRE GROUP	(0〜3:a,b,i,r)
patch(1):TIMBRE NUMBER	(0〜63)
patch(2):KEY SHIFT	(0〜48:−24〜+24)
patch(3):FINE TUNE	(0〜100:−50〜+50)
patch(4):BENDER RANGE	(0〜24)
patch(5):ASSIGN MODE	(0〜3:POLY1〜4)
patch(6):REVERB SWITCH	(0〜1:OFF/ON)

(MT-32マニュアルp35,CM-64マニュアルp30参照)

zm_mt32_print(port,size,id,mes)

機能 MT-32の画面へのメッセージ出力

引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int size:メッセージ長(1〜20)
int id:d0-d15=DEVICE ID(0〜127,-1:current)
str mes:文字列

戻り値 0:正常終了
0以外:エラーコード

備考
メッセージは20文字以内でstr mesに文字列を記述しておく。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_u220_setup(port,id,setup)

機能 U220のセットアップパラメータの設定

引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0〜127,-1:current)
dim char(int) setup(6):セットアップパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char setup()にセットアップパラメータを記述しておく。
dim char setup()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

setup(0):Chorus SW	(0〜1:OFF/ON)
setup(1):Reverb SW	(0〜1:OFF/ON)
setup(2):Rx Cntrl Channel	(1〜17:1〜16,OFF)
setup(3):Patch Change	(0〜5:MAP1〜MAP4,Dir,OFF)
setup(4):Timbre Change	(0〜5:MAP1〜MAP4,Dir,OFF)
setup(5):Rhythm Change	(0〜5:MAP1〜MAP4,Dir,OFF)
setup(6):R.Inst Assign	(0〜5:MAP1〜MAP4,Dir,OFF)

(U220マニュアルp46,p146〜148参照)

注意 U220用の命令はU20に対しても使用できる。以下同様。

zm_u220_part_setup(port,pt_id,setup)

機能 U220の(テンポラリパッチ)パートセットアップパラメータの設定

引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int pt_id:d0-d15=DEVICE ID(0〜127,-1:current)
d16-31=PART NUMBER(1〜6)
dim char(int) setup(12):
パートセットアップパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char setup()にパートセットアップパラメータを記述しておく。
dim char setup()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

setup(0):Timbre Number	(1〜128)
setup(1):Voice Reserve	(0〜30)
setup(2):Receive Channel	(1〜17:1〜16,OFF)
setup(3):Key Range Low	(0〜127:C-1〜G9)
setup(4):Key Range High	(0〜127:C-1〜G9)
setup(5):Velo Level	(0〜1:Above/Below)
setup(6):Velo Threshold	(0〜127)
setup(7):Output Assign	(0〜4:Dry,Rev,Cho,Dir1,Dir2)
setup(8):Level	(0〜127)
setup(9):Pan	(0〜15:L7〜M〜R7)
setup(10):Rx Volume	(0〜1:OFF/ON)
setup(11):Rx Pan	(0〜1:OFF/ON)
setup(12):Rx Hold	(0〜1:OFF/ON)

(U220マニュアルp58,p149参照)

zm_u220_common(port,id,common)

機能 U220の(テンポラリパッチ)コモンパラメータの設定

引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0〜127,-1:current)
dim char(int) common(17):コモンパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char common()にコモンパラメータを記述しておく。
dim char common()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

common(0):Chorus Type	(0〜4:Chorus1,Chorus2,FB-Chorus,Flanger,Short Delay)
common(1):Chorus Out Mode	(0〜1:Pre Rev,Post Rev)
common(2):Chorus Level	(0〜31)
common(3):Chorus Delay	(0〜31)
common(4):Chorus Rate	(0〜31)
common(5):Chorus Depth	(0〜31)
common(6):Chorus Feedback	(1〜63:−31〜0〜+31)
common(7):Reverb Type	(0〜7:Room1〜3,Hall1〜2,GateDelay,CrossDelay)
common(8):Reverb Time	(0〜31)
common(9):Reverb Level	(0〜31)
common(10):Reverb Delay Feedback	(0〜31)
common(11):Reverb Pre Delay Feedback	(0〜31)
common(12):Param1 #	(0〜63:0〜5,7〜31,64〜95,OFF)
common(13):Param1 Param	0:Timbre Level,1:Env Attack 2:Env Decay,3:Env Sustain 4:Env Release,5:A.BendDepth 6:A.Bend Rate,7:Detune Depth 8:Vib Rate,9:Vib Wave Form 10:Vib Depth,11:Vib Delay 12:Vib Rise Time,13:Vib Mod Depth 14:Chorus Level,15:Chorus Rate 16:Chorus Feedback,17:ReverbLevel 18:Delay Feedback)
common(14):Param2 #	(Same as Param1 #)
common(15):Param2 Param	(Same as Param1 Param)
common(16):Param3 #	(Same as Param1 #)
common(17):Param3 Param	(Same as Param1 Param)

(U220マニュアルp54,p149参照)

zm_u220_timbre(port,tm_id,timbre)

機能 U220のティンバーパラメータの設定

引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
int tm_id:d0-d15=DEVICE ID(0〜127,-1:current)
d16-d31=PROGRAM NUMBER(1〜128)
dim char(int) timbre(25):ティンバーパラメータ格納配列

戻り値 0:正常終了
0以外:エラーコード

備考
dim char timbre()にティンバーパラメータを記述しておく。

dim char timbre()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
timbre(0):Tone Media              (0～31:I,1～31)
timbre(1):Tone Number             (1～128)
timbre(2):Timbre Level            (0～127)
timbre(3):Velocity Sens           (1～15:－7～＋7)
timbre(4):Channel Press Sens      (1～15:－7～＋7)
timbre(5):Env Attack Rate         (1～15:－7～＋7)
timbre(6):Env Decay Rate          (1～15:－7～＋7)
timbre(7):Env Sustain Level       (1～15:－7～＋7)
timbre(8):Env Release Rate        (1～15:－7～＋7)
timbre(9):Pitch Shift Coarse      (8～56:－24～＋24)
timbre(10):Pitch Shift Fine       (14～114:－50～＋50)
timbre(11):Bend Range Lower       (0～15:－36,－24,－12～0)
timbre(12):Bend Range Upper       (0～12)
timbre(13):Channel After Sens     (0～27:－36,－24,－12～＋12)
timbre(14):Poly After Sens        (0～27:－36,－24,－12～＋12)
timbre(15):Auto Bend Depth        (0～27:－36,－24,－12～＋12)
timbre(16):Auto Bend Rate         (0～15)
timbre(17):Detune Depth           (0～15)
timbre(18):Rate                   (0～63)
timbre(19):Waveform               (0～8)
timbre(20):Depth                  (0～15)
timbre(21):Delay                  (0～15)
timbre(22):Rise Time              (0～15)
timbre(23):Modulation Depth       (0～15)
timbre(24):Ch After Sens          (0～15)
timbre(25):Poly After Sens        (0～15)
```

(U220マニュアルp65,p149参照)

zm_u220_drums_setup(port,id,setup)

機能　U220の(テンポラリパッチ)パッチドラムセットアップ設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int id:d0-d15=DEVICE ID(0～127,-1:current)
　　　dim char(int) setup(6):
　　　　ドラムセットアップパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char setup()にドラムセットアップパラメータを記述しておく。dim char setup()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
setup(0):Rhythm Setup #           (0～3:1～4)
setup(1):Voice Reserve            (0～30)
setup(2):Receive Channel          (1～17:1～16,OFF)
setup(3):Level                    (0～127)
setup(4):Level Boost Sw           (0～1:OFF/ON)
setup(5):Rx Volume                (0～1:OFF/ON)
setup(6):Rx Hold                  (0～1:OFF/ON)
```

(U220マニュアルp63,p149参照)

zm_u220_drums_inst(int port,int dr_id,char *inst)

機能　U220の(テンポラリパッチ)ドラム音色設定

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int dr_id:d0-d15=DEVICE ID(0～127,-1:current)
　　　　:d16-d31=NOTE NUMBER(35～99)
　　　dim char(int) inst(19):ドラム音色パラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char inst()にドラム音色パラメータを記述しておく。
dim char inst()の構成は必ず下に示す書式に従う。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

```
inst(0):Tone Media                (0～31:I,1～31)
inst(1):Tone Number               (1～128)
inst(2):Source Key                (0～127:C－1～G9)
inst(3):Mute Inst                 (34～98:OFF,B1～D7)
inst(4):Inst Level                (0～31)
inst(5):Velocity Sens             (0～15)
inst(6):Env Mode                  (0～1:Sustain,No Sustain)
inst(7):Env Attack Rate           (1～15:－7～＋7)
inst(8):Env Decay Rate            (1～15:－7～＋7)
inst(9):Env Release Rate          (1～15:－7～＋7)
inst(10):Pitch Shift Coarse       (0～27:－36,－24,－12～＋12)
inst(11):Pitch Shift Fine         (14～114:－50～＋50)
inst(12):Channel After Sens       (0～27:－36,－24,－12～＋12)
inst(13):Poly After Sens          (0～27:－36,－24,－12～＋12)
inst(14):Random                   (0～15)
inst(15):Auto Bend Depth          (0～27:－36,－24,－12～＋12)
inst(16):Auto Bend Rate           (0～15)
inst(17):Detune Depth             (0～15)
inst(18):Output Assign            (0～3:Dry,Rev,Cho,Dir1)
inst(19):Pan                      (0～15:L7～M～R7)
```

(U220マニュアルp72,p149)

zm_u220_print(port,size,id,mes)

機能　U220の画面へのメッセージ出力

引数　int port:MIDIポートの番号(0～3)
　　　　-1はカレントMIDI-OUTを使用
　　　int size:メッセージ長(1～12)
　　　int id:d0-d15=DEVICE ID(0～127,-1:current)
　　　str mes:文字列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

メッセージは12文字以内でstr mesに文字列を記述しておく。
DEVICE IDを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。

zm_m1_setup(setup)

機能　M1のSEQ-SONG0のMIDIチャンネルの設定

引数　dim char(int) setup(7):8パート分のMIDIチャンネル列格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char setup()には必ず8パート分のMIDIチャンネルを記載しておく。

```
setup(0):パート1のMIDIチャンネル
setup(1):パート2のMIDIチャンネル
setup(2):パート3のMIDIチャンネル
setup(3):パート4のMIDIチャンネル
setup(4):パート5のMIDIチャンネル
setup(5):パート6のMIDIチャンネル
setup(6):パート7のMIDIチャンネル
setup(7):パート8のMIDIチャンネル
```

注意　M1用の命令はM1EX/M1R/M1REXに対しても使用できる。以下同様。

zm_m1_part_setup(ptstup)

機能　M1のSEQ-SONG0のトラックパラメータの設定

引数　dim char(int) ptstup(39):
　　　　8パート分のトラックパラメータ格納配列

戻り値　0:正常終了
　　　0以外:エラーコード

備考

dim char ptstup()には必ず8パート分のトラックパラメータを記載しておく。

```
PART 1
ptstup(0):PROGRAM NUMBER          (0～199:I00～C99)
ptstup(1):OUTPUT LEVEL            (0～99)
ptstup(2):KEY TRANSPOSE           ($F4～$0C:－12～12)
ptstup(3):DETUNE                  ($CE～$32:－50～＋50)
ptstup(4):PAN                     (0～13:10:0～0:10,C,C+D,D)

PART 2～8(param(5)～param(39))以下同様の書式構成
```

(M1マニュアルp126参照)

zm_m1_effect_setup(effect)

機能　M1のSEQ-SONG0のエフェクトパラメータの設定

引数　dim char(int) effect(24):エフェクトパラメータ格納配列

戻り値　0:正常終了
0以外:エラーコード

備考

dim char effect()にはダミーデータも含めて必ず25個のエフェクトパラメータが記載されていなければならない。

```
effect(0):Effect 1 Pattern No.          (0～33:1～32,Thru)
effect(1):Effect 2 Pattern No.          (0～33:1～32,Thru)
effect(2):Effect 1 L-Ch E.Balnc         (0～100)
effect(3):Effect 1 R-Ch E.Balnc         (0～100)
effect(4):Effect 2 L-Ch E.Balnc         (0～100)
effect(5):Effect 2 R-Ch E.Balnc         (0～100)
effect(6):Output 3 Pan             (0～101:OFF,100:0～0:100)
effect(7):Output 4 Pan             (0～101:OFF,100:0～0:100)
effect(8):Effect I/O  (BIT0:Effect 1 L-Ch 0=OFF/1=ON
                       BIT1:Effect 1 R-Ch 0=OFF/1=ON
                       BIT2:Effect 2 L-Ch 0=OFF/1=ON
                       BIT3:Effect 2 R-Ch 0=OFF/1=ON)
                       BIT4:Effect 2 0=PARALLEL/1=SERIAL)
effect(9～16):Effect 1 Parameter
effect(17～24):Effect 2 Parameter
```

Structure of Effect Parameter (ofs=9 or 17)

1～3:Hall (4,5:Room,6:Live Stage)

```
effect(ofs+0):Reverb Time (0～97:0.2～9.9),(0～48:0.2～5.0)
effect(ofs+1):Dummy                    0
effect(ofs+2):High Damp                (0～99)
effect(ofs+3):Pre Delay                (0～200)
effect(ofs+4):E/R Level                (0～99)
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

7～9:Early Reflection 1,2,3

```
effect(ofs+0):E/R Time                 (0～70:100～800)
effect(ofs+1):Pre Delay                (0～200)
effect(ofs+2):Dummy                    0
        :                              :
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

10:Stereo Delayt,11:Cross Delay

```
effect(ofs+0):Delay Time L (L)         (0～500)
effect(ofs+1):Delay Time L (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp                (0～99)
effect(ofs+4):Delay Time R (L)         (0～500)
effect(ofs+5):Delay Time R (H)
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

12～13:Stereo Chorus 1～2 (14～15:Flanger)

```
effect(ofs+0):Depth                    (0～99)
effect(ofs+1):Speed           (0～99:0.03～3.00 0.03step
                               100～199:3.1～13.0 0.1step
                               200～216:14.0～30.0 1.0step)
effect(ofs+2):LFO Status     (BIT0:Waveform ＝0:Sin,＝1:Tri
                              BIT1:Phase ＝0:0.0°,＝1:180°
                              BIT2:Wave Shape ＝0:Normal
                                              ＝1:forFlanger)
effect(ofs+3):Feedback             ($9D～$63:－99～＋99)
effect(ofs+4):Delay Time               (0～200),(0～50)
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

16～17:Phase Shifter 1～2

```
effect(ofs+0):Depth                    (0～99)
effect(ofs+1):Speed           (0～99:0.03～3.00 0.03step
                               100～199:3.1～13.0 0.1step
                               200～216:14.0～30.0 1.0step)
effect(ofs+2):LFO Status     (BIT0:Waveform ＝0:Sin,＝1:Tri
                              BIT1:Phase ＝0:0.0°,＝1:180°
                              BIT2:Wave Shape ＝0:Normal
                                              ＝1:forFlanger)
effect(ofs+3):Feedback             ($9D～$63:－99～＋99)
effect(ofs+4):Manual                   (0～99)
effect(ofs+5):Dummy                    0
effect(ofs+6):Dummy                    0
effect(ofs+7):Dummy                    0
```

18～19:Stereo Toremolo 1～2

```
effect(ofs+0):Depth                    (0～99)
effect(ofs+1):Speed           (0～99:0.03～3.00 0.03step
                               100～199:3.1～13.0 0.1step
                               200～216:14.0～30.0 1.0step)
effect(ofs+2):LFO Status     (BIT0:Waveform ＝0:Sin,＝1:Tri
                              BIT1:Phase ＝0:0.0°,＝1:180°
                              BIT2:Wave Shape ＝0:Normal
                                              ＝1:forFlanger)
effect(ofs+3):Shape                ($9D～$63:－99～＋99)
effect(ofs+4):Dummy                    0
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

20:Equalizer

```
effect(ofs+0):Dummy                    0
        :                              :
effect(ofs+3):Dummy                    0
effect(ofs+4):Low fc           (0～2:0.25k,0.50k,1.00k)
effect(ofs+5):High fc                  (0～2:1k,2k,4k)
effect(ofs+6):High Gain            ($F4～$0C:－12～＋12)
effect(ofs+7):Low Gain             ($F4～$0C:－12～＋12)
```

21:Overdrive

```
effect(ofs+0):Dummy                    0
effect(ofs+1):Dummy                    0
effect(ofs+2):Drive                    (0～99)
effect(ofs+3):Level                    (0～99)
effect(ofs+4):Dummy                    0
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

22:Distortion

```
effect(ofs+0):Dummy                    0
effect(ofs+1):Dummy                    0
effect(ofs+2):Distortion               (0～99)
effect(ofs+3):Level                    (0～99)
effect(ofs+4):Dummy                    0
effect(ofs+5):Dummy                    0
effect(ofs+6):Dummy                    0
effect(ofs+7):EQ Low Gain          ($F4～$0C:－12～＋12)
```

23:Exciter

```
effect(ofs+0):Blend                ($9D～$63:－99～＋99)
effect(ofs+1):Emphatic Point           (0～9:1～10)
effect(ofs+2):Dummy                    0
        :                              :
effect(ofs+5):Dummy                    0
effect(ofs+6):EQ High              ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low               ($F4～$0C:－12～＋12)
```

24:Synphonic Ensamble

```
effect(ofs+0):Depth                  (0～99)
effect(ofs+1):Dummy                  0
      :                              :
effect(ofs+5):Dummy                  0
effect(ofs+6):EQ High                ($F4～$0C:－12～＋12)
effect(ofs+7):EQ Low                 ($F4～$0C:－12～＋12)
```

25:Roteffect Speaker

```
effect(ofs+0):Depth                  (0～99)
effect(ofs+1):Dummy                  0
effect(ofs+2):Speed Rate             ($F6～$0A:－10～＋10)
effect(ofs+3):Dummy                  0
      :                              :
effect(ofs+7):Dummy                  0
```

26:Delay/Hall

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):Reverb Time            (0～97:0.2～9.9)
effect(ofs+5):Dummy                  0
effect(ofs+6):High Damp              (0～99)
effect(ofs+7):Pre Delay              (0～150)
```

27:Delay/Room

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):Reverb Time            (0～97:0.2～9.9)
effect(ofs+5):Dummy                  0
effect(ofs+6):High Damp              (0～99)
effect(ofs+7):Pre Delay              (0～150)
```

28:Delay/Early Reflection

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):E/R Time               (0～30:100～400)
effect(ofs+5):Pre Delay              (0～150)
effect(ofs+6):Dummy                  0
effect(ofs+7):Dummy                  0
```

29:Delay/Chorus

```
effect(ofs+0):Delay Time L (L)       (0～500)
effect(ofs+1):Delay Time L (H)
effect(ofs+2):Feedback L             ($9D～$63:－99～99)
effect(ofs+3):High Damp L            (0～99)
effect(ofs+4):Delay Time R (L)       (0～500)
effect(ofs+5):Delay Time R (H)
effect(ofs+6):Feedback R             ($9D～$63:－99～99)
effect(ofs+7):High Damp R            (0～99)
```

30:Delay/Chorus (31:Delay Flanger)

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):Depth                  (0～99)
effect(ofs+5):Speed              (0～99:0.03～3.00 0.03step
                                 100～199:3.1～13.0 0.1step
                                 200～216:14.0～30.0 1.0step)
effect(ofs+6):LFO Status (BIT0:Waveform ＝0:Sin,＝1:Tri
                          BIT1:Phase ＝0:0.0°,＝1:180°
                          BIT2:Wave Shape ＝0:Normal
                                          ＝1:for Flanger)
effect(ofs+7):Feedback               0,($9D～$63:－99～99)
```

32:Delay/Phaser

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):Depth                  (0～99)
effect(ofs+5):Speed                  (0～99:0.03～3.00 0.03step)
effect(ofs+6):Feedback               ($9D～$63:－99～99)
effect(ofs+7):Dummy                  0
```

33:Delay/Tremolo

```
effect(ofs+0):Delay Time (L)         (0～500)
effect(ofs+1):Delay Time (H)
effect(ofs+2):Feedback               ($9D～$63:－99～99)
effect(ofs+3):High Damp              (0～99)
effect(ofs+4):Depth                  (0～99)
effect(ofs+5):Speed                  (0～99:0.03～3.00 0.03step)
effect(ofs+6):Dummy                  0
effect(ofs+7):Shape                  ($9D～$63:－99～99)
```

(M1マニュアルp127参照)

zm_m1_print(size,mes)

機能　M1のSEQ-SONG0のタイトル設定
引数　int size:メッセージ長(1～10)
　　　str mes:文字列
戻り値　0:正常終了
　　　0以外:エラーコード
備考
　メッセージは10文字以内でstr mesに文字列を記述しておく。

zm_send_to_m1(port,id)

機能　M1へパラメータを送信する
引数　int port:MIDIポートの番号(0～3)
　　　　　　-1はカレントMIDI-OUTを使用
　　　int id:d0-d15=DEVICE ID(0～127,-1:current)
戻り値　0:正常終了
　　　0以外:エラーコード
備考
　DEVICE ID＝&H30＋(グローバルチャンネル－1)
　グローバルチャンネルはGLOBALモードのF5－1で設定できる。
　idは省略可能。省略すると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
　必ず'm1_setup()'～'m1_print()'を設定してからこの命令を実行すること。
　'm1_effect_setup()'のみ省略が可能。このときはドライバのデフォルトデータが送信される。
　'm1_setup()'～'m1_print()'の間にM1関係以外のコマンドがあると正常なデータが送信されない。

例

```
    dim char a(8)={2,1,4,3,6,5,8,7};
     /*音色,音量,ｷｰﾄﾗﾝｽﾎﾟｰｽﾞ,ﾃﾞﾁｭｰﾝ,ﾊﾟﾝのﾌｫｰﾏｯﾄ*/
     dim char b(40)={0,99,0,0,5,  /*TRACK 1*/
                     0,99,0,0,5,  /*TRACK 2*/
                     0,99,0,0,5,  /*TRACK 3*/
                     0,99,0,0,5,  /*TRACK 4*/
                     0,99,0,0,5,  /*TRACK 5*/
                     0,99,0,0,5,  /*TRACK 6*/
                     0,99,0,0,5,  /*TRACK 7*/
                     0,99,0,0,5}; /*TRACK 8*/
     dim char c(25)={&H21,&H21};
     m1_setup(a)
     m1_part_setup(b)
     m1_effect_setup(c)
     m1_print(7,"Zenji.N")
     send_to_m1(-1,&H30)
```

zm_pcm_read(num,flag,type,orig,param)

機能　PCMファイルの登録
引数　int num:登録番号 (ノート番号の場合:0～32767)

(音色番号の場合:$8000+(0〜32767)
int flag:PCM加工コマンド(PPC)でPCMデータ加工するか
(-1:YES,0:NO)
int type:登録次元(-1:AD PCM,1=16BitPCM,0=ver.2.0互換)
int orig:オリジナルキー:0〜127
dim char(int) param(〜):登録データ情報およびPPC列格納配列
(備考参照)

戻り値 0以外:dim char param()処理後のアドレス
0:エラー

備考

dim char param()からはまず、最大32文字分の音色名文字列を格納する。終端コードは0。

次にディスクからPCMファイルを読みこむ場合はそのファイル名を格納する。終端コードは0。...(1)

ディスクからデータを読み込まずにすでにZ-MUSIC側に登録されているPCMデータをコピーして加工するには、音色名の後ろに $0000 を格納しさらにその後ろに参照番号を指定する。...(2)

PPCでPCMを加工する場合には、PPC列をこの後ろに格納しておく。

また、PPCでPCMを加工する場合は必ずflagを－1とすること。flagが0ではたとえPPCがあっても無視される。

PPCについてはドキュメントMEASURE12を参照。

(1)の場合

[TONE NAME...最大32文字],0,[FILENAME文字列],0,PPCコマンド列...

(2)の場合

[TONE NAME...最大32文字],0,$00,$00,参照先番号(2bytes),PPCコマンド列...

参照先番号は、
ノート番号の場合:0〜32767
音色番号の場合:$8000+(0〜32767)
で表される。

zm_pcm_erase(num)

機能 PCMファイルの登録取消
引数 int num:取消したい登録番号(ノート番号の場合:0〜32767)
(音色番号の場合:$8000+(0〜32767)
戻り値 0:正常終了
0以外:エラーコード

zm_register_zpd(zpd)

機能 ZPDの登録
引数 str zpd:登録するZPDのファイルネーム
戻り値 0:正常終了
0以外:エラーコード

zm_set_zpd_table(ver,zpd)

機能 ZPDテーブルの登録
引数 int ver:ZPDバージョンタイプ
(0:ver.1.0〜ver.2.0,1:ver.3.0,2:PDX)
dim char(int) zpd(〜):
登録するZPDテーブルを格納している配列(ZPDヘッダ含まず)
戻り値 0:正常終了
0以外:エラーコード

備考

ZPDのテーブルの構造についてはMEASURE12を参照

zm_exec_subfile(filename)

機能 サブファイルの読み込みと実行
引数 str filename:サブファイルのファイル名
戻り値 0:正常終了
0以外:エラーコード

備考

サブファイルとは演奏の際に必要となる別ファイルにまとめた補助ファイルのことを指す。具体的には、演奏ファイル(ZMS,ZMD,OPM)、AD PCM定義リスト(CNF),ZPDファイル、MIDIデータファイル(MDD,MID)など。

zm_transmit_midi_dump(port,filename)

機能 MIDIデータファイルの転送
引数 int port:MIDIポートの番号(0〜3)
-1はカレントMIDI-OUTを使用
str filename:MIDIデータファイルのファイル名
戻り値 0:正常終了
0以外:エラーコード

備考

MIDIデータファイルには、ver.1.0からver.2.0で標準MIDIデータファイルとされてきたMDDファイルと、ver.3.0から採用されたスタンダードMIDIファイル(FORMAT0)のMIDIファイルの2タイプがあるが、この関数ではどちらも楽器へ転送することができる。

zm_set_wave_form1(wave,data)

機能 波形メモリの登録1
引数 int wave:波形番号(8〜32767)
dim char(int) data(〜):波形メモリデータ格納配列
戻り値 0:正常終了
0以外:エラーコード

備考

dim char data()で表される波形メモリデータは以下の構成を取る。

data(0〜3)(4bytes):波形のデータサイズ(必ず偶数)
data(4):ループタイプ(0:oneshot,1:repeat,2:altanative)
data(5〜8)(4bytes):ループ開始オフセット
data(9〜12)(4bytes):ループ終了オフセット
data(13〜16)(4bytes):ループ回数(0:∞,1〜)
data(17〜20)(4bytes):ダミー
data(21):波形名文字列長
data(22〜)(?bytes):文字列(終端の0なし)
偶数要素(アドレス)から実際の波形データ(signed short)
:
:

※本関数で登録された波形はZ-MUSICに取り込まれるため、関数実行後、波形データ本体のメモリを解放しても構わない。

zm_set_wave_form2(wave,data)

機能 波形メモリの登録2
引数 int wave:波形番号(8〜32767)
dim char(int) data(〜):波形メモリデータ格納配列
戻り値 0:正常終了
0以外:エラーコード

備考

dim char data()で表される波形メモリデータの構成は、zm_set_wave_form1()と同じ。

zm_set_wave_form2()では、波形データがZ-MUSICに取り込まれないため、登録後も波形データは保存される必要がある。

zm_obtain_events(omt,event)

機能 各種イベントの取得
引数 int omt:どのイベントに関する設定を行うかのビットマーカー
ビット=1でそのイベントの設定を更新する。
dim int(char) event(7):各種イベント発生時のジャンプ先と
ジャンプ条件が記述された配列。
構成は備考参照。
-1とするとint omtによらず設定は一切行わない。
戻り値 現在の設定を格納した領域のアドレス。構成は備考参照。

備考

event(0):m_playが実行されたときのジャンプ先アドレス omtのd0ビット
event(1):m_stopが実行されたときのジャンプ先アドレス omtのd1ビット
event(2):m_contが実行されたときのジャンプ先アドレス omtのd2ビット
event(3):演奏が終了したときのジャンプ先アドレス omtのd3ビット
event(4):指定ループ回数(0:OFF,1〜256) omtのd4ビット
event(5):指定回数ループしたときのジャンプ先アドレス omtのd5ビット
event(6):指定クロック(0:OFF,1〜) omtのd6ビット
event(7):演奏開始後指定クロック経過したときの
ジャンプ先アドレス omtのd7ビット

int omtでビット=1としたパラメータのみをint event(0)から記述しておく。int omtでビット=0としたパラメータに関しては記述しない。

たとえばint omt=$81の場合、event(0)に、
「m_playが実行されたときのジャンプ先アドレス」
を、event(1)に、
「演奏開始後指定クロック経過したときのジャンプ先アドレス」
を記述しておく。

zm_get_loop_time()
機能　現在の演奏のループ回数を取得する
引数　なし
戻り値　-1以外:現在のループ回数(1〜)
　　　-1:エラー

zm_mask_tracks(track)
機能　トラックマスク
引数　dim int(char) track(〜):マスクするトラックとマスクモードのペアを格納した配列
戻り値　設定できなかったトラック番号(short int)の並びを格納した領域のアドレス
　　　終端コード=-1(short)
備考
dim int track()からは、
　　トラック番号(0〜65534)*65536
　　+マスクモード(0:MASK OFF,1:REVERSE,-1:MASK ON)
の数値列を記載する。終端コードは(short)-1。

zm_mask_all_tracks(mode)
機能　全トラックマスク
引数　int mode:(0:MASK OFF,1:REVERSE,-1:MASK ON)
戻り値　0:正常終了
　　　0以外:エラーコード

zm_solo_track(track)
機能　ソロトラック演奏
引数　int(char) track:トラック番号(0〜65534)
戻り値　0:正常終了
　　　0以外:エラーコード

zm_mask_channels(channel)
機能　チャンネルマスク
引数　dim int(char) channel(〜):マスクするチャンネルIDとマスクモードのペアを格納した配列
戻り値　設定できなかった(int)チャンネルIDの並び,終端コード=(int)-1
備考
dim int channel()からは、
　チャンネルID(32ビット値),マスクモード(16ビット値)
の数値列を記載する。
なお、チャンネルIDは、
　チャンネルID(上位16ビット):0=FM音源,1=AD PCM音源
　　$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
　　-1=カレントMIDI
　チャンネルID(下位16ビット):チャンネル番号0〜15
マスクモードは(0:MASK OFF,1:REVERSE,-1:MASK ON)の値をとる。
終端コードは-1。
たとえばMIDI1の15チャンネル(ID:$8000_000e)をマスク(ID:$ffff)、AD PCMの3チャンネル(ID$0001_0002)をマスク反転(ID:$0001)する場合は、
　{ &h8000000e,&hffff0001,&h00020001,-1}
となる。

zm_mask_all_channels(mode)
機能　全チャンネルマスク
引数　int mode:(0:MASK OFF,1:REVERSE,-1:MASK ON)
戻り値　0:正常終了
　　　0以外:エラーコード

zm_solo_channel(channel)
機能　ソロチャンネル演奏
引数　int channel:(上位16ビット):0=FM音源,1=AD PCM音源
　　$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
　　-1=カレントMIDI
　int channel:(下位16ビット):チャンネル番号0〜15
戻り値　0:正常終了
　　　0以外:エラーコード

zm_set_ch_output_level(out)
機能　各チャンネルの出力レベルの設定
引数　dim char(int) out(8):出力パラメータを記載した配列
戻り値　0:正常終了
　　　0以外:エラーコード
備考
dim char out()には以下の構成の出力パラメータを格納しておく。
out(0〜1)(2bytes):デバイス(0=FM音源,1=AD PCM音源
　　$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
　　-1=カレントMIDI)
out(2〜3)(2bytes):チャンネル番号(0〜15)
out(4):パラメータ有効ビット(d0:移動速度,d1:開始ﾚﾍﾞﾙ,d2:終了ﾚﾍﾞﾙ)
　(対応ビット=1で、そのパラメータが後続する)
out(5〜6)(2bytes):移動速度(フェーダー移動速度。0で動かさない)
out(7):フェーダー移動開始レベル(0〜128)(128が原音ﾚﾍﾞﾙ)
out(8):フェーダー移動終了レベル(0〜128)(128が原音ﾚﾍﾞﾙ)

zm_set_tr_output_level(out)
機能　各トラックの出力レベルの設定
引数　dim char(int) out(6):出力パラメータを記載した配列
戻り値　0:正常終了
　　　0以外:エラーコード
備考
dim char out()には以下の構成の出力パラメータを格納しておく。
out(0〜1)(2bytes):トラック番号(0〜65534)
out(2):パラメータ有効ビット(d0:移動速度,d1:開始ﾚﾍﾞﾙ,d2:終了ﾚﾍﾞﾙ)
　(対応ビット=1で、そのパラメータが後続する)
out(3〜4)(2bytes):移動速度(フェーダー移動速度。0で動かさない)
out(5):フェーダー移動開始レベル(0〜128)(128が原音ﾚﾍﾞﾙ)
out(6):フェーダー移動終了レベル(0〜128)(128が原音ﾚﾍﾞﾙ)

zm_master_fader(fdr)
機能　マスターフェーダーの設定
引数　dim char(int) fdr(6):出力パラメータを記載した配列
戻り値　0:正常終了
　　　0以外:エラーコード
備考
dim char fdr()には以下の構成の出力パラメータを格納しておく。
out(0〜1)(2bytes):デバイス(0=FM音源,1=AD PCM音源
　　$8000=MIDI1,$8001=MIDI2,$8002=MIDI3
　　-1=カレントMIDI)
out(2):パラメータ有効ビット(d0:移動速度,d1:開始ﾚﾍﾞﾙ,d2:終了ﾚﾍﾞﾙ)
　(対応ビット=1で、そのパラメータが後続する)
out(3〜4)(2bytes):移動速度(フェーダー移動速度。0で動かさない)
out(5):フェーダー移動開始レベル(0〜128)(128が原音ﾚﾍﾞﾙ)
out(6):フェーダー移動終了レベル(0〜128)(128が原音ﾚﾍﾞﾙ)

zm_get_fader_status()
機能　フェーダーの状態取得
引数　なし
戻り値　d0,d1,d6,d7ビットに検査結果が反映される。
　d0:実行中のチャンネルフェーダーすべてが目的に達している
　　(1:達成した 0:達成していない)
　d1:実行中のマスターフェーダーすべてが目的に達している
　　(1:達成した 0:達成していない)
　d6:チャンネルフェーダーの状態(1:実行中 0:実行されていない)
　d7:マスターフェーダーの状態(1:実行中 0:実行されていない)

zm_get_play_time()
機能　演奏時間の取得
引数　なし
戻り値　16進表現で00HHMMSSという書式で返ってくる。
　なおHH:0〜99,MM:0〜59,SS:0〜59の値を取る。

zm_get_1st_comment()
機能　演奏中の曲のコメントを取り出す
引数　なし
戻り値　文字列アドレス
　　　終端コードは0

zm_get_timer_mode()
機能　現在のテンポソースとなっているタイマの種類を返す

引数　　なし
戻り値　0:タイマ A,1:タイマ B,2:YM3802タイマ

zm_get_track_table()
機能　　演奏トラックテーブルのアドレスを得る
引数　　なし
戻り値　演奏トラックテーブルのアドレス
備考
　演奏トラックテーブルには演奏に使用されるトラック番号(2bytes)が-1(2bytes)を終端コードとし羅列される。

zm_get_track_table_se()
機能　　効果音演奏トラックテーブルのアドレスを得る
引数　　なし
戻り値　効果音演奏トラックテーブルのアドレス
備考
　効果音演奏トラックテーブルには演奏に使用されるトラック番号(2bytes)が-1(2bytes)を終端コードとし羅列される。

zm_get_play_work(track)
機能　　演奏トラックワークのアドレスを得る
引数　　int track:トラック番号(0〜65534)
戻り値　演奏トラックワークのアドレス
備考
　演奏トラックワークの内容と構成についてはMEASURE13を参照のこと。

zm_get_play_work_se(track)
機能　　効果音演奏トラックワークのアドレスを得る
引数　　int track:トラック番号(0〜65534)
戻り値　効果音演奏トラックワークのアドレス
備考
　効果音演奏トラックワークの内容と構成は通常の演奏トラックとまったく同一。

zm_get_buffer_information()
機能　　バッファアドレステーブルのアドレス
引数　　なし
戻り値　バッファアドレステーブルのアドレス
備考
　バッファアドレステーブルの構成についてはMEASURE13を参照のこと

zm_zmsc_status()
機能　　Z-MUSICの状態ワークのアドレス
引数　　なし
戻り値　Z-MUSICの状態ワークのアドレスの構成はMEASURE13を参照のこと
備考
　各ワークの構成についてはMEASURE13を参照のこと

zm_calc_total(zmd,result)
機能　　ZMDの演奏時間やトータルステップタイムなどの計算
引数　　dim char zmd(〜):ZMDを格納した配列
　　　　　　　　　　　　　　　(ヘッダはあってもなくてもよい)
　　　　int result:戻り値を格納するための変数
戻り値　0:正常終了
　　　　0以外:発生したエラーの数
備考
　戻り値が0の場合はint resultには計算結果データ列を格納した領域のポインタが返される。計算結果データ列の構成についてはMEASURE10ファンクションZM_CALC_TOTALの項を参照のこと。
　戻り値が0以外の場合はint resultにはエラーの内容を格納したエラーテーブルのアドレスが返される。エラーテーブルの構成についてはMEASURE11を参照のこと。
　いずれの場合もint resultに返された領域は使用後、zm_free_mem()などで解放すること。
　この関数はコンパイラに含まれる外部組み込みのファンクションとであるため、使用前にはコンパイラの常駐を確認する必要がある。

zm_occupy_zmusic(mode)
機能　　Z-MUSICの占有
引数　　int mode:1=占有する,0=占有を解除する,-1=占有状態の検査
戻り値　mode=0,1のとき:以前の占有状態(mode=-1のときを参照)
　　　　mode=-1のとき:0=占有されていない,1=占有されている

zm_occupy_compiler(mode)
機能　　コンパイラの占有
引数　　int mode:1=占有,0=占有解除,-1=占有状態検査
戻り値　mode=0,1のとき:以前の占有状態(mode=-1のときを参照)
　　　　mode=-1のとき:0=占有されていない,1=占有されている

zm_store_error(err,noferr,addr)
機能　　エラーコードのストア
引数　　int err(上位16):エラーの発生したファンクション番号
　　　　　　　　　　　　　　　　　　　　　　(0〜32767)
　　　　　　(下位16):エラーコード(0〜65535)
　　　　　int err=-1とした場合は問い合わせのみでストアは行わない
　　　　int noferr:エラーの合計個数値に幾つ加算するか
　　　　int addr:エラーストックバッファ先頭アドレスを格納する変数
戻り値　現在ストックしているエラーの個数
備考
　本関数はユーザーアプリケーション側で発生したエラーをZ-MUSIC側に知らせたい場合に使用する。ただしその場合、エラーコードフォーマットはMEASURE11で示してある形式に従わなければならない。

zm_print_error(mode,lang,noferr,ZMD,srcaddr,srcname,errtbl,buff)
機能　　エラーメッセージの表示
引数　　int mode:メッセージの出力先(0:画面,1:バッファ)
　　　　int lang:メッセージの言語(0:英語,1:日本語)
　　　　int noferr:エラーの数
　　　　str zmd:ZMDのファイル名(ない場合はzmd="")
　　　　dim char(int) srcaddr(〜):ソース(ZMS)のアドレス
　　　　　　　　　　　　　　　　(ない場合はsrcaddr(0)=0)
　　　　str srcname:ソース(ZMS)ファイル名(ない場合はsrcname="")
　　　　int errtbl:エラーストックバッファの先頭アドレス
　　　　int buff:出力バッファの先頭アドレスを格納する変数
　　　　　　　(int mode=1のときのみ有効,バッファの終端には0)
戻り値　出力バッファのサイズ。ただし終端の０は含まない。
備考
　エラーテーブルからエラーメッセージを得る場合などに使用する。zmd〜srcnameは通常の外部アプリケーションならば0やNUL文字列でよい。
　コンパイラなどの場合はそれらを正しく設定すればテキストエディタなどのタグジャンプに対応した出力を得ることができる。

zm_get_mem(size,emply)
機能　　メモリの確保
引数　　int size:確保サイズ
　　　　int emply:確保するメモリの用途ID
戻り値　0:確保に失敗した
　　　　0以外:確保したメモリブロックのアドレス
備考
　メモリの用途IDはMEASURE10ファンクションZM_GET_MEMの項を参照のこと。
　Z-MUSICに無関係のメモリブロックを確保する場合ならば当関数を用いる必要はない。

zm_enlarge_mem(size,addr)
機能　　メモリブロックのサイズの変更
引数　　int size:変更後のサイズ
　　　　int addr:zm_get_mem()で確保したメモリのアドレス
戻り値　確保しなおしたメモリブロックのアドレス
8　　　 ０ならば確保に失敗した事を意味する
備考
　メモリブロックのサイズを大きくした場合などはメモリブロックの再配置が自動で行われる。そのためint addrで与えられたメモリブロックが別のアドレスに移動する場合がある。よってこの関数を実行後はint addrを戻り値のアドレス値に置き換えるべきである。
　変更に失敗した場合は、int addrのメモリブロックの内容は保証される。

zm_free_mem(addr)
機能　　メモリブロックの解放

引数　int addr:zm_get_mem()で確保したメモリのアドレス
戻り値　0:解放成功
　　　0以外:エラーコード

zm_free_mem2(emply)
機能　特定の用途IDを持ったメモリブロックの解放
引数　int emply:メモリの用途ID
戻り値　0:解放成功
　　　0以外:エラーコード
備考
　メモリの用途IDはMEASURE10 ファンクションZM_GET_MEMの項を参照のこと。

zm_exchange_memid(mode,s_emply,d_emply,addr)
機能　メモリブロックの用途IDの変更
引数　int mode:変更モード(0:チェックなしで変更,
　　　　　　1:チェックしてから更新
　　　　　　-1:int s_emplyの用途IDを持つ
　　　　　　　メモリブロックをすべてint d_emplyの
　　　　　　　用途IDへ変更する)
　　　int s_emply:更新元メモリの用途ID(int mode=0の場合は
　　　　　　意味なし)
　　　int d_emply:希望メモリの用途ID
　　　int addr:zm_get_mem()で確保したメモリのアドレス
戻り値　0:変更成功
　　　0以外:エラーコード
備考
　メモリの用途IDはMEASURE10 ファンクションZM_GET_MEMの項を参照のこと。

zm_init_all()
機能　Z-MUSICの全初期化を行う
引数　なし
戻り値　Z-MUSICのバージョンID
備考
　通常のアプリケーションは本関数を利用する必要はないバージョンIDはたとえばバージョン3.21ならば $00003241となる。4の部分は固定値。

zm_int_start(tmtype)
機能　割り込み開始制御
引数　int tmtype:制御対象タイマタイプ
　　　　　(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)
戻り値　なし
備考
　通常のアプリケーションは本関数を利用する必要はない。

zm_int_stop(tmtype)
機能　割り込み停止制御
引数　int tmtype:制御対象タイマタイプ
　　　　　(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)
戻り値　なし
備考
　通常のアプリケーションは本関数を利用する必要はない。

zm_control_tempo(mode)
機能　テンポ設定の主導権設定
引数　int mode:-1=テンポ設定の主導権を外部アプリケーションに委託
　　　　　0=テンポ設定の主導権をZ-MUSICに戻す
　　　　　(このとき内部パラメータでテンポの初期化も実施する)
戻り値　0:設定正常終了
　　　0以外:設定失敗

zm_convert_pcm(mode,size,source,destination)
機能　PCMデータの変換を行う
引数　int mode:変換モード(0:AD PCM to PCM,1:PCM TO AD PCM)
　　　int size:変換元のPCMのデータサイズ
　　　dim char(or int) source(～):変換元となる(AD)PCMデータの
　　　　　　格納配列
　　　dim char(or int) destination(～):変換先となる(AD)PCMデータ
　　　　　　の格納配列
戻り値　なし

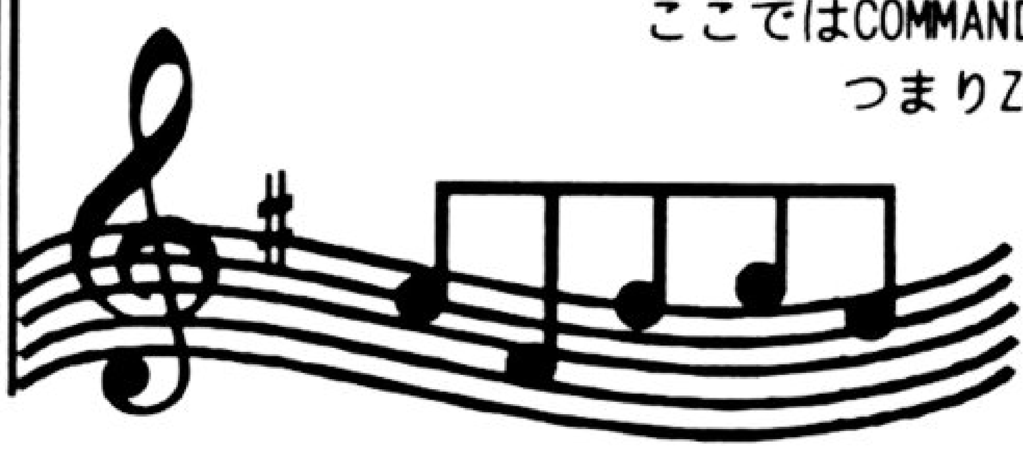
zm_work(track,offset)
機能　演奏トラックワークの内容取得
引数　int track:トラック番号(0～65534)
　　　int offset:演奏トラックワークオフセット
戻り値　ワーク内容(0～255)
備考
　演奏トラックワークの内容と構成についてはMEASURE13を参照のこと。

zm_exec_zmd(track,size,zmd)
機能　ZMD列の実行
引数　int track:トラック番号(0～65534,65535:特に指定しない場合)
　　　int size:ZMD列の長さ
　　　dim char zmd:ZMD列格納配列
　　　　　(配列自体はsize+8の大きさを必要とする)
戻り値　なし

MEASURE 4

ZMSコマンド

ここではCOMMAND.X上（コマンドモード）から音楽プログラムを書くための書式、
つまりZMSファイル（OPMファイル）の文法について解説します。



4．1．　ZMSファイルとは

Z-MUSICシステムではエディタからMMLやコマンドを記述したファイルを演奏することができる。これをZ-MUSICシステムでは「ZMSファイル」と呼ぶ。X68000本体に付属している「OPMDRV.X」用の「OPMファイル」と概念的には、ほとんど同じものである。

ZMSファイルはED.Xなどのスクリーンエディタを使って、ここで解説する「ZMSコマンド」を記述することによって作成していく。ZMSコマンドのなかでもトラックに依存しないZMSコマンドを特に「共通コマンド」と呼んでいる。

4．2．　ZMSコマンド

[注意]

・省略してもよいパラメータはその旨を記述してあるが、それ以外は省略不可である。

・コマンドは大文字小文字どちらで記述しても構わないが、ここでは説明の都合上コマンド名を大文字、パラメータを小文字で記述している。

・'*'をつけたものはパラメータを複数行にわたって記述できる。それ以外のコマンドではその1行ですべてのパラメータを記述しなければならない。

・MUSICZ3.FNCの外部関数命令と対応するものはそのコマンド名を挙げてある。

■初期化

●Z-MUSICと音源の初期化

(I)
.INITIALIZE

Z-MUSICと音源の初期化を行う。通常はZMS先頭に書く。なお、Ver.3.0では、このZMSコマンドを記述しなくても最低限の初期化は演奏時に行われるので特に記述する必要はなくなっている。

(MEASURE3 zm_init()参照)

例

```
.INITIALIZE
```

■各種モード設定

●マスタークロックの設定

(Zn)
.MASTER_CLOCK n

・n: 1～32767

全音符の絶対音長nを設定する。設定範囲は1≦n≦32767。初期値は192。通常は設定する必要はない。この値を大きくすればするほどシステムへの負荷は大きくなるが精細な音長表現が可能になる。逆に小さくすればするほどシステムの負荷は軽くなるが短い音長を使用した場合に誤差を生じることがある。

1920, 480, 192, 144, 128, 96といった数値を設定するのが一般的である。たとえばn=1920として全音符の絶対音長を1920とした場合、4分音符は1920/4=480で表される。

(MEASURE3 zm_set_master_clock()参照)

例

```
.MASTER_CLOCK 480
```

●ジャンプ系コマンドの制御

.JUMP md

・md:DISABLE,ENABLE

ジャンプ系MMLコマンドの生成を抑制する。

md=DISABLEとした場合はこの生成を抑止し、md=ENABLEとした場合はこの生成を行う。抑制対象となるMMLコマンドは以下のジャンプ系MML。

[@]	全トラック追従ジャンプ
[!]	単一トラックジャンプ
[end]	演奏停止
[JUMP]	小節ジャンプ

例

```
.JUMP DISABLE
```

●ゲートタイム解像度の設定

.GATETIME_RESOLUTION n

・n:8,16,32,64,128

MML Qの最大値を設定する。設定値は8,16,32,64,128のいずれか。初期設定時は8になっている。大きな値に設定すればより細かなゲートタイム指定が行えるようになる。

例

```
.GATETIME_RESOLUTION 128
```

●音長指定方式の選択

.LENGTH_MODE md

・md:MML,STEP

音符の解釈方式を選択する。モードは「MMLモード」と「STEPモード」がある。

MMLモードは、いわゆる一般的なMML方式で、音符を、

<音符[変化記号]> [音長] [発音長] [ベロシティ]

(<>は省略不可、[]は省略可のパラメータを表している)

の順で記述するもの。音長、および発音長は、全音符を1とした音楽的音長で指定する。

一方、STEPモードでは、ステップ入力型DTMツールの音符入力方式を模倣したもので、

<キー>[変化記号][オクターブ] [ステップタイム] [ゲートタイム] [ベロシティ]

のような書式で音符を記述する。ステップタイムはMMLモードの音長に相当し、ゲートタイムはMMLモードの発音長に相当する。ただし、両方ともMMLモードでいうところの絶対音長で表記する点が異なる(STEPモードでは使用できる音長が絶対音長のみとなる)。

たとえばオクターブ3の4分音符(絶対音長=48)のド#をベロシティ100、発音長を符点8分音符相当(絶対音長=36)で鳴らしたい場合、MMLモードでは、

```
o3 c#4,8.,100
```

となるが、STEPモードでは、

```
c#3,48,36,100
```

と記述する。

例

```
    .LENGTH_MODE STEP
```

●相対ベロシティMMLの指定

.RELATIVE_VELOCITY md

・md:_￣,￣_,@U,U

初期状態では_￣は相対音量のMMLであるが、これを相対ベロシティに効果を変換することができる。本命令はこの変換についての設定を行うもの。

md=_￣あるいはmd=￣_とすると、相対ベロシティは _￣で行うことができるようになる。一方、md=@Uまたはmd=Uとすると相対ベロシティは従来どおり@U±n,U±nで指定することになり、_￣は相対音量の操作MMLとして機能する。

例

```
    .RELATIVE_VELOCITY _￣
    .RELATIVE_VELOCITY @U
```

●FM音源チューニング

.FM_TUNE_SETUP {t1,t2,...,t128} ＊

・t1〜t128:-32768〜+32768

FM音源のチューニング設定を 0-1C(C-1) から 09G(G9) まで128個のノート個別に行う(概念的にはノート個別に設定するディチューンのようなもの)。

t1〜t128はそれぞれノート番号0(C-1)から127(G9)までに対応する。

t1〜t128の設定範囲は-32768〜+32767で、64が半音の変化量に相当する。ディチューン設定を行うMML @Kに与える数値と同次元。

t1〜t128はt1から任意の個数まで設定したあと設定を打ち切ることができる。その場合、省略されたものについては0が設定されたと見なされる。

例

```
        .fm_tune_setup {0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1,9,-10,9,-8
                        0,-5,3,10,-9,-1,-6,1}
```

（Cを主音とした平均律の例）

●AD PCM音源チューニング

.ADPCM_TUNE_SETUP {t1,t2,...,t128} ＊
.PCM_TUNE_SETUP {t1,t2,...,t128} ＊

・t1〜t128:-32768〜+32768

AD PCM(PCM)音源のチューニング設定を 0-1C(C-1) から 09G(G9) まで128個のノート個別に行う。

t1〜t128はそれぞれノート番号0(C-1)から127(G9)までに対応する。

t1〜t128の設定範囲は-32768〜+32767で、64が半音の変化量に相当する。ディチューン設定を行うMML @Kに与える数値と同次元。

t1〜t128はt1から任意の個数まで設定したあと設定を打ち切ることができる。その場合、省略されたものについては０が設定されたと見なされる。

例

```
        .pcm_tune_setup{-4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3,-30,0,-6,-31
                        -4,29,-1,-8,-33,-5,28,-3}
```

（アラビア音階）

■拍子/調号

●拍子設定

.METER m/n

.m/n:1/1〜255/255

拍子の設定を行う。ただし、この設定は演奏になんら影響を与えるものではなく、あくまで演奏データに対するひとつの付加情報的な意味あいしか持たない。

m/nは音楽的な拍子をそのまま分数表現で与えることができる。

例

```
        .METER  3/4
              （4分の3拍子を設定）
```

●調号設定

.KEY { n,type,m }
.KEY str

・n:０〜７

・type:-,+,#

・m:MAJOR,MINOR

・str:文字列

調号の設定を行う。この設定は演奏になんら影響を与えるものではなくあくまで、演奏データに対するひとつの付加情報的な意味あいしか持たない。

nは調号中の変化記号の個数で設定範囲は0〜7。

typeは調号に含まれる変化記号の個数を表し、設定項目は－，＋，＃のいずれか。－は♭を表し、＋または＃は＃を表す。

mは長調か短調のいずれかを選択するパラメータで、設定項目はMAJOR,MINORのいずれか。MAJORが長調、MINORが短調に対応する。

また、一般的な調は以下に示す英語表記の調名にて指定することもできる。

```
    Cmajor,Gmajor,Dmajor,Amajor,Emajor,Bmajor,F+major,F#major,
    C+major,C#majorFmajor,B-major,Bbmajor,E-major,Ebmajor,A-major,
    Abmajor,D-major,Dbmajor,G-major,Gbmajor,C-major,Cbmajor
    Aminor,Eminor,Bminor,F+minor,F#minor,C+minor,C#minor,G+minor,
    G#minor,D+minor,D#minor,A+minor,A#minor,Dminor,Gminor,Cminor,
    Fminor,B-minor,Bbminor,E-minor,Ebminor,A-minor,Abminor
```

例

```
        .KEY C#minor              /嬰ハ短調
```

●演奏時間

.PERFORMANCE_TIME hh:mm:ss

hh:00〜99

mm:00〜59

ss:00〜59

演奏時間を設定する。この設定は演奏になんら影響を与えるものではなくあくまで、演奏データに対するひとつの付加情報的な意味あいしか持たない。

hh,mm,ssはそれぞれ、時間、分、秒を表し、00:00:01〜99:59:59までの値を取る。

例

```
        .PERFORMANCE_TIME 00:01:32
```

■トラック定義/チャンネルアサイン

●トラック定義/チャンネルアサイン

(A dev,trk,trkv,trkf,trks,trkm,cmnt)
.ASSIGN trk {dev,trkv,trkf,trks,trkm,cmnt} ＊

・trk:1〜65535

・dev:FM1〜FM8,MIDI1-1〜MIDI4-16,ADPCM1〜16

・trkv:０〜256

・trkf:１〜256

・trks:PLAY,RECORD

・trkm:NORMAL,RHYTHM

・cmnt:文字列

演奏で使用するトラックの定義を行う。

・trk:トラック番号

trkはトラック番号で1～65535与えることができる。
・dev:デバイス名
devはそのトラックをどの音源の演奏に用いるかを指定するもの。FM1～FM8はFM音源チャンネル1～8,ADPCM1～ADPCM16はAD PCMチャンネル1～16に対応する。またAD PCMはPCM1～PCM16としても指定できる。デバイス名のFMやAD PCMの後ろに'-'を添付しFM-1～FM-8,ADPCM-1～ADPCM-16のように定することも可能。
MIDIはデバイス名MIDIの後ろにインタフェイス番号,チャンネル番号を順番に記述して指定する。インタフェイス番号を省略することもできる。省略時は後述のカレントMIDI-OUTとして選択されているインタフェイスを指定したことになる。
・trkv:トラックボリューム
そのトラックの出力音量を指定する。設定範囲は0～256で、0が無音状態、128が原音量に相当する。trkvは省略可能で省略時は128が設定される。
・trkf:トラック処理頻度
そのトラックの処理頻度を指定する。設定範囲は1～256。たとえば2を指定するとそのトラックは2回に1回の割合でしか処理されず、システムに対する負荷を軽くすることができる。ただし、短い音長を使用した音符を演奏すると演奏に誤差を生じる場合がある。
trkfは省略可能で省略時は1が設定される。
・trks:トラック状態
そのトラックの使用目的を指定する。設定項目はPLAY,RECORDのいずれか。ただし、現在指定できるのはPLAY(演奏)のみ。trksは省略可能で省略時はPLAYが設定される。
・trkm:トラックモード
そのトラックの演奏処理形式を指定する。設定項目はNORMALまたはRHYTHM。
NORMALはノートオン/ノートオフを通常に処理する演奏モード。
RHYTHMに設定するとノートオフ処理を省略する演奏モードで処理される。
RHTHM指定時はちょうどMMLで@R1を設定したときと同じになる。
trkmは省略可能で省略時はNORMALが設定される。
・cmnt:コメント
そのトラックに与える注釈を添付することができる。
文字列は"～"で括る必要はなく、何文字でも可能。
(MEASURE3 zm_assign()参照)
例

```
.ASSIGN 1 FM1
(A MIDI10,10)
.ASSIGN 10 {MIDI2-1,128,1,PLAY,NORMAL,MELODY LINE}
(A PCM-1,1,128,2,PLAY,RHTHM,DRUM PART)
```

■演奏制御

●演奏開始

(Pn1,n2,…,ni)
.PLAY {n1,n2,…,ni} ＊

・n1～ni: 1～65535
指定されたトラック(n1～ni)の演奏を開始する。n1～niはトラック番号を表し、設定範囲は1≦ni≦65535。全パラメータを省略した場合は全トラックの演奏を開始する。
(MEASURE3 zm_play()参照)
例

```
(P)
.PLAY
(P1,2,3)
.PLAY {
        1,2,3,4
        5,6,7,8
      }
```

●演奏停止

(Sn1,n2,…,ni)
.STOP {n1,n2,…,ni} ＊

・n1～ni: 1～65535
指定されたトラック(n1～ni)の演奏を停止する。n1～niはトラック番号を表し、設定範囲は1≦ni≦65535。全パラメータを省略した場合は全トラックの演奏を停止する。
(MEASURE3 zm_stop()参照)
例

```
(S)
.STOP
(S1,2,3)
.STOP {
        1,2,3,4
        5,6,7,8
      }
```

●演奏再開

(Cn1,n2,…,ni)
.CONTINUE {n1,n2,…,ni} ＊

・n1～ni: 1～65535
指定されたトラック(n1～ni)の演奏を再開する。n1～niはトラック番号を表し、設定範囲は1≦ni≦65535。全パラメータを省略した場合は全トラックの演奏を再開する。
(MEASURE3 zm_cont()参照)
例

```
(C)
.CONTINUE
(C1,2,3)
.CONTINUE {
        1,2,3,4
        5,6,7,8
      }
```

●トラックマスク

.TRACK_MASK { trk1,mode1, trk2,mode2, ... trkX,modeX } ＊

・trkX: 1～65535,ALL
・modeX:OFF,ON,REVERSE
演奏中の任意のトラックをリアルタイムにマスク/解除する。
trkXは操作対象とするトラック番号を表し、1～65535を設定する。また、演奏中すべての演奏トラックを操作対象とする場合は文字列 ALL を指定する。
modeXはtrkXで指定したトラックの演奏状態をどう変更するかを指定する。
以下にmodeXの設定項目とその効果を示す。

modeX	効果
OFF	そのトラックをマスクする
ON	そのトラックのマスクを解除する
REVERSE	そのトラックのマスクを反転する

(MEASURE3 zm_mask_tracks()参照)
例

```
.TRACK_MASK {1,ON
             2,OFF
             3,REVERSE
}
```

■フェーダー制御

●マスターフェーダー

.MASTER_FADER {dev1,spd1,st1,ed1,dev2,spd2,st2,ed2,...,devX,spdX,stX,edX} ＊

・devX:ALL,FM,AD PCM,MIDI1,MIDI2,MIDI3
・spdX: 0～255
・stX: 0～128
・edX: 0～128
マスターフェーダーはZ-MUSICの演奏の最終的な出力制御を司るフェーダーで、後述のトラック単位のトラックフェーダーとは独立して制御が行える。
本命令は、デバイスdevXのマスターフェーダーを速度spdXでレベルstXからedXまで移動させるもの。
devXは操作対象デバイスで、ALL,FM,AD PCM,MIDI1,MIDI2,MIDI3のいずれかを指定する。ALLでは全デバイスのマスターフェーダーを操作対象とする。
spdXはフェーダーの移動速度で数値が大きいほど高速に移動する。また、spdX=0では「静止」の指定となる。

stXはフェーダー移動の開始レベルに相当し、0が無音、128が最大(原音)レベルに対応する。
edXはフェーダー移動の終了レベルに相当し、0が無音、128が最大(原音)レベルに対応する。
(MEASURE3 zm_master_fader()参照)
例

```
.MASTER_FADER {ALL,10,128,0}
.MASTER_FADER {
        MIDI1,10,128,0
        MIDI2,10,0,128
        MIDI3,0,64,64
}
```

●トラックフェーダー
.TRACK_FADER{trk1,spd1,st1,ed1,trk2,spd2,st2,ed2, ...,trkX,spdX,stX,edX}　＊

・trkX: 1～65535
・spdX: 0～255
・stX: 0～128
・edX: 0～128
トラックフェーダーはトラック単位にその演奏出力制御が行えるフェーダーである。
本命令は、トラックtrkXのトラックフェーダーを速度spdXでレベルstXからedXまで移動させるもの。
trkXは操作対象トラック番号で、設定範囲は1～65535。
spdXはフェーダーの移動速度で数値が大きいほど高速に移動する。また、spdX=0では「静止」の指定となる。
stXはフェーダー移動の開始レベルに相当し、0が無音、128が最大(原音)レベルに対応する。
edXはフェーダー移動の終了レベルに相当し、0が無音、128が最大(原音)レベルに対応する。
(MEASURE3 zm_set_ch_output_level(),zm_set_tr_output_level()参照)
例

```
.TRACK_FADER {
        1,10,128,0
        2,10,0,128
        3,0,64,64
}
```

■テンポ

●テンポ
(Ot)
.TEMPO t

・t: 1～32767
テンポをtに設定する。1分間にt個の4分音符を演奏するようになる。tの設定範囲は1～32767。ただし、テンポ制御タイマにタイマAを使用した場合のtの設定範囲は77-32767となる。
(MEASURE3 zm_tempo()参照)
例

```
(o120)
.TEMPO 150
```

■MMLセット

●MMLセット
(Tn1,n2,…,n8)
.TRACK n1,n2,…,n8 { MML }　＊

・n1～n8: 1～65535
MMLをトラックn1～n8へセットする。設定範囲は1≦n1～n8≦65535。トラック番号を複数指定した場合は同一MMLを同時に複数のトラックにセットする。また、n2以降は省略可能。
MMLをセットするトラックはあらかじめ.ASSIGN命令で宣言しておかなければならない。
なお、MMLについてはMEASURE 5を参照のこと。
例

```
.TRACK 1,2,3 {
c4 d4 e4
}
```

●パターントラック
.PATTERN name,trkf { MML }　＊

・name:文字列
・trkf: 1～256
パターントラックを定義する。パターントラックとは曲中頻繁に使うリズムパターンなどを固有の名前を与えて定義し、曲中任意に呼び出すことができる演奏トラックのこと。具体的には、本命令でパターントラックを定義し、MML、
　　[PATTERN]
　　[EMBED]
で曲中で呼び出す。
nameはパターン名を表し、これは251文字以内の英数字文字列で指定する。
trkfは.ASSIGN命令のtrkfに相当するもので、トラック処理頻度を表す。設定範囲は1～256で、省略可能。省略時は1が設定される。
MMLについてはMEASURE 5を参照のこと。
例

```
.PATTERN FUNK,1 {
        o2 @u99 c4 @u120 d4 @u99c8 c8 @u127 d4
}
```

■FM音源の音色設定

●FM音源の音色設定
(Vn,0,v1,v2,…,v55)　＊

・n:1～32768
・v1～v55:下記参照
FM音源の音色設定を行う。
nは定義する音色番号で1≦n≦32768。
v1～v55の音色パラメータについては下記参照。

●音色フォーマット(OPMDRV.X互換方式)

v1:AF(ﾌｨｰﾄﾞﾊﾞｯｸ/ｱﾙｺﾞﾘｽﾞﾑ)		(0～63)
v2:OM(ｽﾛｯﾄﾏｽｸ)		(0～15)
v3:WF(ｳｪｰｳﾞﾌｫｰﾑ)		(0～3)
v4:SYC(ｼﾝｸﾛ)		(0,1)
v5:SPD(ｽﾋﾟｰﾄﾞ)		(0～255)
v6:PMD		(0～127)
v7:AMD		(0～127)
v8:PMS		(0～7)
v9:AMS		(0～3)
v10:PAN		(0～3)
v11:DUMMY		
v12:OP1, v13:OP2, v14:OP3, v15:OP4	AR(ｱﾀｯｸﾚｰﾄ)	(0～31)
v16:OP1, v17:OP2, v18:OP3, v19:OP4	1DR(1stﾃﾞｨｹｲﾚｰﾄ)	(0～31)
v20:OP1, v21:OP2, v22:OP3, v23:OP4	2DR(2ndﾃﾞｨｹｲﾚｰﾄ)	(0～31)
v24:OP1, v25:OP2, v26:OP3, v27:OP4	RR(ﾘﾘｰｽﾚｰﾄ)	(0～15)
v28:OP1, v29:OP2, v30:OP3, v31:OP4	1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)	(0～15)
v32:OP1, v33:OP2, v34:OP3, v35:OP4	TL(ﾄｰﾀﾙﾚﾍﾞﾙ)	(0～127)
v36:OP1, v37:OP2, v38:OP3, v39:OP4	KS(ｷｰｽｹｰﾘﾝｸﾞ)	(0～3)
v40:OP1, v41:OP2, v42:OP3, v43:OP4	MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)	(0～15)
v44:OP1, v45:OP2, v46:OP3, v47:OP4	DT1(ﾃﾞｨﾁｭｰﾝ1)	(0～7)
v48:OP1, v49:OP2, v50:OP3, v51:OP4	DT2(ﾃﾞｨﾁｭｰﾝ2)	(0～3)
v52:OP1, v53:OP2, v54:OP3, v55:OP4	AME(AMSｲﾈｰﾌﾞﾙ)	(0,1)

※1DRをDR、2DRをSR、1DLをSL、TLをOL、MULをMLと表記する場合もある。
(MEASURE3 zm_vset()参照)
例

```
 (v1,0
/       AF  OM  WF SYC SPD PMD AMD PMS AMS PAN
        60, 15,  2,  0,210, 40,  0,  2,  0,  3,  0
/       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
        31,  5,  0, 12,  2, 30,  1,  2,  7,  0,  0
        31,  5,  0, 12,  8,  6,  1,  2,  5,  0,  0
        31,  5,  0, 12,  8, 28,  1,  2,  3,  0,  0
        31,  5,  0, 12,  8,  6,  1,  2,  5,  0,  0)
```

●FM音源の音色設定

(@n,v1,v2,…,v55)　＊
.FM_TIMBRE n,cmnt {v1,v2,…,v55}　＊
.FM_VSET n,cmnt {v1,v2,…,v55}　＊

・n: 1～32768
・cmnt:コメント
・v1～v55:下記参照

AL/FB分離形式によるFM音源の音色設定を行う。
nは定義する音色番号で1≦n≦32768。
cmntには音色に関するコメント文字列を与えることができる。ただし省略可能。
v1～v55の音色パラメータについては下記参照。

●音色フォーマット(AL/FB分離方式)

v1:OP1,	v2:OP2,	v3:OP3,	v4:OP4	AR(ｱﾀｯｸﾚｰﾄ)	(0～31)
v5:OP1,	v6:OP2,	v7:OP3,	v8:OP4	1DR(1stﾃﾞｨｹｲﾚｰﾄ)	(0～31)
v9:OP1,	v10:OP2,	v11:OP3,	v12:OP4	2DR(2ndﾃﾞｨｹｲﾚｰﾄ)	(0～31)
v13:OP1,	v14:OP2,	v15:OP3,	v16:OP4	RR(ﾘﾘｰｽﾚｰﾄ)	(0～15)
v17:OP1,	v18:OP2,	v19:OP3,	v20:OP4	1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)	(0～15)
v21:OP1,	v22:OP2,	v23:OP3,	v24:OP4	TL(ﾄｰﾀﾙﾚﾍﾞﾙ)	(0～127)
v25:OP1,	v26:OP2,	v27:OP3,	v28:OP4	KS(ｷｰｽｹｰﾘﾝｸﾞ)	(0～3)
v29:OP1,	v30:OP2,	v31:OP3,	v32:OP4	MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)	(0～15)
v33:OP1,	v34:OP2,	v35:OP3,	v36:OP4	DT1(ﾃﾞｨﾁｭｰﾝ1)	(0～7)
v37:OP1,	v38:OP2,	v39:OP3,	v40:OP4	DT2(ﾃﾞｨﾁｭｰﾝ2)	(0～3)
v41:OP1,	v42:OP2,	v43:OP3,	v44:OP4	AME(AMSｲﾈｰﾌﾞﾙ)	(0,1)

v45:AL(ｱﾙｺﾞﾘｽﾞﾑ)	(0～7)
v46:FB(ﾌｨｰﾄﾞﾊﾞｯｸ)	(0～7)
v47:OM(ｽﾛｯﾄﾏｽｸ)	(0～15)
v48:PAN	(0～3)
v49:WF(ｳｪｰｳﾞﾌｫｰﾑ)	(0～3)
v50:SYC(ｼﾝｸﾛ)	(0,1)
v51:SPD(ｽﾋﾟｰﾄﾞ)	(0～255)
v52:PMD	(0～127)
v53:AMD	(0～127)
v54:PMS	(0～7)
v55:AMS	(0～3)

(MEASURE3 zm_vset()参照)

例

```
.FM_TIMBRE 1,SQUARE WAVE {
        /      AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
               31,  0,  2,  0,  0, 21,  0,  1,  0,  0,  0
               31,  0,  0,  8,  0,  3,  0,  3,  0,  0,  0
               31,  0,  0,  8,  0,  3,  0,  1,  0,  0,  0
               31,  0,  0,  8,  0,  3,  0,  1,  0,  0,  0
        /      AL  FB  OM PAN  WF SYC SPD PMD AMD PMS AMS
                5,  7, 15,  3   0,  0,  0,  0,  0,  0,  0}
```

■AD PCM/PCMデータ加工/登録

●PCM音色の登録

.8BITPCM_TIMBRE b,t,k,str { filename,PPC }　＊
.16BITPCM_TIMBRE b,t,k,str { filename,PPC }　＊
.ADPCM_TIMBRE b,t,k,str { filename,PPC }　＊

・b: 0～255
・t: 1～128
・k:0-1C～09G または C-1～G9
・str:文字列
・filename:文字列
・PPC:後述

filenameで指定されたPCMデータを音色(TIMBRE)として登録する。.8BITPCM_TIMBREは8BIT PCM、.16BITPCM_TIMBREは16BIT PCM,.ADPCM_TIMBREはAD PCMをそれぞれ登録する。
b,tはそれぞれ登録先バンク番号，登録先音色番号を表し、設定範囲は0≦b≦255,1≦t≦128。
kは登録するPCMデータの原音高(オリジナルキー)を設定するもので設定範囲は0-1C～09G または C-1～G9。Z-MUSICがPCMの音程を変化させる場合は、このkの設定をもとに行うことになる。
strはコメントで登録する音色についての簡単な注釈文などを添えることができる(省略可能)。
filenameを、読み込むファイルの名前でなく、

　　.TIMBRE b,t
　　.TONE b,n

といった書式で指定すれば、以前に、

.TIMBRE b,tならば.8BITPCM_TIMBRE,.16BITPCM_TIMBRE,.ADPCM_TIMBRE
.TONE b,tならば.8BITPCM_TONE,.16BITPCM_TONE,.ADPCM_TONE

を用いて登録した音色/トーンのコピーを用いて加工登録することもできる。
「.TIMBRE b,t」書式では、bがバンク番号(0～255)、tが音色番号(1～128)を、「.TONE b,n」書式では、bがバンク番号(1～256)、nがトーン番号(0～127)を表す。nは0-1C～09G または C-1～G9といった音階MMLで指定することもできる。
PPC(PCM PROCESSING COMMAND)はfilenameで指定されたPCMデータを登録する前に加工処理を行う命令で、次項のようなものがある。各PPCは指定された順序に従い加工を実行していく。PPCは省略、あるいは、同種PPCの複数設定も可能。
(MEASURE6参照)

例

```
.16BITPCM_TIMBRE 0,10,02C,BASS  {SLAP.P16,.VOLUME 120}
.ADPCM_TIMBRE 1,10,02C,BASS2 {.TIMBRE 0,10,.BEND 15600,22000}
```

本命令でバンクb、音色番号tで登録したPCMは[PCM_MODE TIMBRE]にしたトラックでMML Ｉでバンク番号b、MML @で音色番号を指定すると使用することができる。

例

```
.ADPCM_TIMBRE 1,10,02C,BASS2 {.TIMBRE 0,10,.BEND 15600,22000}
```

で登録した音色は、

```
(t1)[PCM_MODE TIMBRE] i0,1 @10
```

で、またはMML @のみで、

```
(t1)[PCM_MODE TIMBRE] @1:10
```

と指定することもできる。

●PCMトーンの登録

.8BITPCM_TONE t,k,str { filename,PPC }　＊
.16BITPCM_TONE t,k,str { filename,PPC }　＊
.ADPCM_TONE t,k,str { filename,PPC }　＊

・t: 1～256
・k: 0～127 あるいは 0-1C～09G または C-1～G9
・str:文字列
・filename:文字列
・PPC:後述

filenameで指定されたPCMデータをトーン(TONE)として登録する。
.8BITPCM_TONEは8BIT PCM、.16BITPCM_TONEは16BIT PCM,.ADPCM_TONEはAD PCMをそれぞれ登録する。
tは登録先トーンセット番号を表し、設定範囲は1≦t≦256。
kは登録するノート番号を指し示すもので設定範囲は0～127あるいは、音階MMLを用いて0-1C～09G または C-1～G9と指定することもできる。
strはコメントで登録する音色についての簡単な注釈文などを添えることができる(省略可能)。
filenameを、読み込むファイルの名前でなく、

　　.TIMBRE b,t
　　.TONE b,n

といった書式で指定すれば、以前に。

.TIMBRE b,tならば.8BITPCM_TIMBRE,.16BITPCM_TIMBRE,.ADPCM_TIMBRE
.TONE b,tならば.8BITPCM_TONE,.16BITPCM_TONE,.ADPCM_TONE

を用いて登録した音色/トーンのコピーを用いて加工登録することもできる。
「.TIMBRE b,t」書式では、bがバンク番号(0～255)、tが音色番号(1～128)を、「.TONE b,n」書式では、bがバンク番号(1～256)、nがトーン番号(0～127)を表す。nは0-1C～09G または C-1～G9といった音階MMLで指定することもできる。
PPC(PCM PROCESSING COMMAND)はfilenameで指定されたPCMデータを登録する前に加工処理を行う命令で、次項のようなものがある。各PPCは指定された順序に従い加工を実行していく。PPCは省略、あるいは、同種PPCの複数設定も可能。
(MEASURE6参照)

PCM PROCESSING COMMAND

●音程の変更(その1)

.PITCH k,o,s

・k:0-1C〜09G,C-1〜G9,-144〜+144
・o:0〜4294967295
・s:0〜4294967295

音高を半音単位で変更する
・k:変更先音高
kで音程変更先の音高を音階MML(0-1C〜09G,C-1〜G9)で指定する。
相対数値(-144〜+144)で指定した場合は原音高からの半音単位の指定となる。
・o:変更開始位置
音高変更をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:変更サイズ
音高変更を何カウント分行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてを変更範囲とする。
例

```
.PITCH 04A,0,1200
.PITCH -8,80,1200
```

●音程の変更(その2)

.TUNE t1,t2,o,s

・t1:1〜65535
・t2:1〜65535
・o:0〜4294967295
・s:0〜4294967295
音高の変更をサンプリング周波数比で指定する。

・t1:変更元サンプリング周波数
・t2:変更先サンプリング周波数
t1が変換元のサンプリング周波数、t2が変更目的のサンプリング周波数。値の単位は[Hz]でt1, t2ともに設定範囲は1〜65535。
・o:変更開始位置
音高変更をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:変更サイズ
音高変更を何カウント分行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてを変更範囲とする。
例

```
.TUNE 15600,22000,0,1200
```

●ポルタメント

.PORTAMENT k,o,s

・k:0-1C〜09G,C-1〜G9,-144〜+144
・o:0〜4294967295
・s:0〜4294967295
ポルタメントを行う。

・k:ポルタメント先の音程
kでポルタメント先の音程を音階MML(0-1C〜09G,C-1〜G9)で指定する。
相対数値(-144〜+144)で指定した場合は原音程からの半音単位の指定となる。
・o:ポルタメント開始位置
ポルタメントをデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:ポルタメント範囲
ポルタメントを何カウント内で行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてをポルタメント範囲とする。
例

```
.PORTAMENT 04A,0,1200
.PORTAMENT -8,80,1200
```

●ピッチベンド

.BEND t1,t2,o,s

・t1:1〜65535
・t2:1〜65535
・o:0〜4294967295
・s:0〜4294967295
ポルタメントをサンプリング周波数次元で実行する。

・t1:ピッチベンド元サンプリング周波数
・t2:ピッチベンド先サンプリング周波数
t1がピッチベンド元のサンプリング周波数、t2がピッチベンド先のサンプリング周波数。値の単位は[Hz]でt1, t2ともに設定範囲は1〜65535。
・o:ピッチベンド開始位置
ピッチベンドをデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:ピッチベンド範囲
ピッチベンドを何カウント内で行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてをピッチベンド範囲とする。
例

```
.BEND 04A,0,1200
.BEND -8,80,1200
```

●音量の変更

.VOLUME v,o,s

・v:0〜65535
・o:0〜4294967295
・s:0〜4294967295
原音量を100とした音量変更を行う。

・v:変更先音量
原音量を100とし、変更目的の音量vを0〜65535の範囲で指定する。0が無音、65535が最大音量に相当する。
・o:変更開始位置
音量変更をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:変更サイズ
音量変更を何カウント分行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてを変更範囲とする。
例

```
.VOLUME 150,480,1800
```

●反転指定

.REVERSE o,s

・o:0〜4294967295
・s:0〜4294967295
逆転を行う。

・o:逆転開始位置
逆転処理をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:変更サイズ
逆転処理を何カウント分行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてを逆転処理する。
例

```
.REVERSE 0,1800
```

●フェードイン/フェードアウト

.FADE v1,v2,o,s

・v1:0〜65535
・v2:0〜65535
・o:0〜4294967295
・s:0〜4294967295
フェードイン/フェードアウトを行う。

・v1:変更元音量
・v2:変更先音量
原音量を100とし変化元音量v1から目的音量v2までの音量変化を線形に行う。v1,v2ともに設定範囲は0〜65535。0が無音、65535が最大音量に相当する。
・o:音量操作開始位置
音量操作をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:音量操作範囲
音量操作を何カウント内で行うかを指定する。設定範囲は0〜4294967295。省略時は、o以降すべてをフェード操作範囲とする。
例

```
.FADE 100,50,480,1800
```

(オフセットカウント480から1800カウント内で音量100から50へ変化させる)

●ディストーション

.DISTORTION v,c,o,s

・v:0〜65535
・c:0〜65535
・o:0〜4294967295
・s:0〜4294967295
ディストーション(歪み)効果を与える。

・v:増幅倍率
原音量を100とし音量をv倍に増幅する。設定範囲は0〜65535。0が無音、65535が最大増幅に相当する。
・c:切り捨て音量レベル
v倍してできた波形に対しのピークレベルc内に音量を丸め込む。
設定範囲は0〜65535。
・o:ディストーション操作開始位置
ディストーション操作をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0〜4294967295。省略時は0が採択される。
・s:ディストーション効果適用範囲
ディストーション操作を何カウント行うかを指定する。設定範囲は0〜4294

4967295。省略時は、o以降すべてをディストーション操作範囲とする
例
.DISTORTION 500,100,480,1800
(オフセットカウント480から1800カウント内でディストーション処理を行う。音量を500%に増幅したものを音量100に丸め込む)

●スムージング
.SMOOTH r,o,s
・r: 1～65535
・o: 0～4294967295
・s: 0～4294967295
スムージング(平滑化)を行う。処理後の音はやや曇った感じになる。

・r:平滑化処理回数
平滑化を連続何回行うかを設定する。設定範囲は1～65535。
・o:スムージング操作開始位置
スムージング操作をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0～4294967295。省略時は0が採択される。
・s:スムージング効果適用範囲
スムージング操作を何カウント行うかを指定する。設定範囲は0～4294967295。省略時は、o以降すべてをスムージング操作範囲とする。
例
.SMOOTH 10,480,1800
(オフセットカウント480から1800カウント内でスムージングを10回行う)

●切り出し
.TRUNCATE o,s
・o: 0～4294967295
・s: 0～4294967295
指定した領域を切り出す。

・o:切り出し開始位置
切り出し操作をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0～4294967295。省略時は0が採択される。
・s:切り出しサイズ
切り出しサイズを何カウントにするかを指定する。設定範囲は0～4294967295。省略時は、o以降すべてを切り出す指定になる。
例
.TRUNCATE 1200　　(オフセットカウント1200以降を切り出す)
.TRUNCATE ,1200　　(先頭から1200カウント切り出す)

●削除
.DELETE o,s
・o: 0～4294967295
・s: 0～4294967295
指定した領域を削除する。

・o:削除開始位置
削除をデータ先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0～4294967295。省略時は0が採択される。
・s:削除サイズ
削除を何カウント行うかを指定する。設定範囲は0～4294967295。省略時は、o以降すべてを削除する指定になる。
例
.DELETE 1200　　(オフセットカウント1200以降を削除する)
.DELETE ,1200　　(先頭から1200カウント削除する)

●合成
.MIX filename,o
(.MIX filename,o,PPC)
・filename:文字列
・o:-2147483648～2147483647
filenameで表されるAD PCM/PCMデータを合成する

・filename:合成データ名
合成するAD PCM/PCMデータのファイル名を設定する。拡張子の省略は不可。また、filenameを合成するファイルの名前でなく、
.TIMBRE b,t
.TONE　b,t
といった書式で指定すれば、以前に、PCMデータの登録コマンドで登録した音色/トーンのコピーを合成することもできる。
「.TIMBRE b,t」書式では、bがバンク番号(0～255)、tが音色番号(1～128)を、「.TONE b,n」書式では、bがバンク番号(0～255)、nがトーン番号(0～127)を表す。nは0-1C～09G または C-1～G9といった音階MMLで指定することもできる。
・o:合成開始位置
合成操作を被合成データの先頭から何カウント目から行うかを設定する。
オフセット値。設定範囲は-2147483648～2147483647。負値を設定した場合は、その絶対値カウント分、合成データの方を先行させて合成する。省略時は0が採択される。
例
.MIX CYMBAL.P16,-1200
.MIX .TONE 4,03D,4800
なお、PPC「.MIX」はその他のPPCと違い(...)で括れば.MIX自体にもPPCを局所的に設定することができる。
例
(.MIX .TIMBRE 10,10,1000,.VOLUME 150,.PITCH -8)
(後続の「V120」「P-8」は合成するデータに対して有効となる。つまりこの例では、TIMBRE 10,10の音量を150%化、8半音分の音程を下げたものを被合成データ先頭から1000カウント後の位置から合成する設定となる)

●接合
.CONNECT filename,o
(.CONNECT filename,o,PPC)
・filename:文字列
・o: 0～4294967295
filenameで表されるAD PCM/PCMデータを接合する。

・filename:接合データ名称
接合するAD PCM/PCMデータのファイル名を設定する。拡張子の省略は不可。また、filenameを接合するファイルの名前でなく、
.TIMBRE b,t
.TONE　b,t
といった書式で指定すれば、以前に、PCMデータの登録コマンドで登録した音色/トーンのコピーを接合することもできる。
「.TIMBRE b,t」書式では、bがバンク番号(0～255)、tが音色番号(1～128)を、「.TONE b,n」書式では、bがバンク番号(0～255)、nがトーン番号(0～127)を表す。nは0-1C～09G または C-1～G9といった音階MMLで指定することもできる。
・o:接合開始位置
接合操作を被接合データの先頭から何カウント目から行うかを設定するオフセット値。設定範囲は0～4294967295。省略時は被接合データの再後尾位置が採択される。PPC「.LOOP」が未設定の場合は接続したデータを無限ループさせるような設定をデフォルトで行う。
例
.CONNECT CYMBAL.P16,-1200
.CONNECT .TONE 4,03D,4800

なお、PPC「.CONNECT」はその他のPPCと違い(...)で括れば.CONNECT自体もPPCを局所的に設定することができる。
例
(.CONNECT .TIMBRE 10,10.VOLUME 150, .PITCH -8)
(後続の「V120」「P-8」は合成するデータに対して有効となる。つまりこの例では、TIMBRE 10,10の音量を150%化、8半音分の音程を下げたものを被合成データの後ろに接合する設定となる)

●ループ領域の設定
.LOOP t,r,o,s
・t:0
・r: 0～4294967295
・s: 0～4294967295
・e: 0～4294967295
ループ領域を設定する。

・t:ループタイプ
ループの仕方を設定する。現在のところ設定できるのは0の順方向(FORWARD)のみ。
・r:繰り返し回数
ループ領域を何回繰り返して再生するかを設定する。0は半永久的に、発音が終了するまで繰り返し続ける設定となる。1～4294967296はその数の回数分繰り返しを行い、ループ領域外の再生に移る。
・s:ループ開始位置
ループ開始ポイントをデータ先頭から何カウント目に設定するかのオフセット値。設定範囲は0～4294967295。省略時は0が採択される。
・e:ループ終了位置
ループ終了ポイントをデータ先頭から何カウント目に設定するかのオフセット値。設定範囲は0～4294967295。省略時は、ループ開始位置s以降すべてをループ領域と設定する指定になる。
例
.LOOP 0,10,480,4000　　(オフセットカウント480からオフセットカウント4000までを順方向に10回繰り返し再生する)

例

```
.16BITPCM_TONE 0,36,BASS  {BD.P16,.VOLUME 120}
.ADPCM_TONE   10,o2d,SNARE {.TIMBRE 0,10,.BEND 15600,22000}
```

本命令でトーンセットt、ノート番号kに登録したPCMは[PCM_TIMBRE TONE]にしたトラックで(デフォルト状態)MML @でトーンセット番号を指定し、希望のノートをキーオンさせることで鳴らすことができる。

例

```
.ADPCM_TONE   10,o2d,SNARE {.TIMBRE 0,10,.BEND 15600,22000}
```

で登録した音色は、

```
(t1)[PCM_MODE TONE] @10 o2 d
```

として演奏させることができる。

TIMBREとTONEの違い

Z-MUSIC ver.2.0ではAD PCMデータの利用法といえば、0～127個の鍵盤に1音1音割り当てて利用するドラムマシンのような使い方しかありませんでした。この方法はドラムパートやパーカッションパートには向いた使用方法でしたが、ひとつの楽器でメロディを演奏するときには、使用する音高の全音を個別に鍵盤ごとに登録する必要があり、非常に面倒なうえ、ZPDが肥大化するという問題がありました。

そこでver.3.0からは1音を原音程とともに登録すれば、希望の音高をZ-MUSIC(MPCM)側でリアルタイムに自動的に演奏してくれるモードが新設されました。

ver.2.0からのドラムマシン的なAD PCM利用方法をTONEモード、そしてver.3.0から新設されたモードをTIMBREモードと呼びます。

TONEモードでは1～256の256個の音色番号(ver2.0では1～4の4個まで)があり、その1つひとつの音色番号に対して0～127の128個のノートごとに任意のAD PCM/PCMデータを割り当てることが可能です。

●登録したPCM音色の削除

.ERASE_TIMBRE b,t

・b:0～255
・t:1～128

登録済みのPCM音色のうち不要なものを削除する。

パラメータb,tはPCM音色登録コマンドの同名パラメータと同じ書式。
bはバンク番号で0～255、tは音色番号で1～128の値を取る。
(MEASURE6参照)

例

```
.16BITPCM_TIMBRE 2,10,02C,BASS  {SLAP.P16,.VOLUME 120}
.ERASE_TIMBRE 2,10
```

●登録したPCMトーンの削除

.ERASE_TONE t,k

・t:1～256
・k:0～127 あるいは 0-1C～09G または C-1～G9

登録済みのPCMトーンのうち不要なものを削除する。

パラメータt,kはPCMトーン登録コマンドの同名パラメータと同じ書式。
tはトーンセット番号で1～256、kはノート番号で0～127、あるいは音階MMLを用いて0-1C～09G または C-1～G9のような指定も可能。。
(MEASURE6参照)

例

```
.ADPCM_TONE 10,o2d,SNARE {.TIMBRE 0,10,.BEND 15600,22000}
.ERASE_TONE 10,o2d
```

●AD PCMデータ定義ファイルの読み込み

.ADPCM_LIST filename
.CNF filename

・filename:文字列

filenameで指定したAD PCMデータの定義ファイル(CNFファイル)を読み込み実行する。filenameで与えるファイルネームは拡張子が省略可能。省略時は.CNFが自動添付され処理される。
(MEASURE3 zm_exec_subfile(), MEASURE6参照)

例

```
.ADPCM_LIST orchestra.cnf
.ADPCM_LIST standard          (standard.CNFが読み込まれる)
```

CNFファイル、ZPDファイルとは

AD PCMデータ登録/削除を行う共通コマンドのみを記述したZMSを特にCNFファイルと呼んでいます。CNFファイルはZ-MUSICシステムに含まれる標準PCMファイルコンバータZPCNV3.RでコンバートするとZPDファイルに変換することができます。

CNFファイルに用いることのできる共通コマンドは以下の通りです。
.16BITPCM_TIMBRE,.16BITPCM_TONE,.8BITPCM_TIMBRE,.8BITPCM_TONE,
.ADPCM_TIMBRE,.ADPCM_TONE,.ADPCM_BANK,.ERASE_TIMBRE,.ERASE_TONE,.ERASE

CNFファイル中に指定されたAD PCMデータに対する加工処理などをあらかじめ行い結果を1個のファイルにまとめたものをZPDファイルと呼びます。このためZPDファイルはCNFファイルに比べて実際のAD PCMデータ登録を高速にまとめて行うことができます。

●ZPDの読み込み

.ADPCM_BLOCK_DATA filename
.ZPD filename

・filename:文字列

filenameで指定されたZPDファイルの読み込みを行う。
filenameの拡張子は省略が可能で、省略時は'.ZPD'が自動添付される。
(MEASURE3 zm_register_zpd(), MEASURE6参照)

例

```
.ADPCM_BLOCK_DATA SFX.ZPD
.ZPD  JAZZ_DRUM               (JAZZ_DRUM.ZPDが読み込まれる)
```

■ver.2.0式AD PCMデータ登録

●AD PCMデータ登録先バンクの設定

.ADPCM_BANK n

AD PCM音の登録先のバンクを指定する。

n=バンク番号。1≦n≦4。
初期化共通コマンド'(I)'命令でデフォルト値1が設定される。
(MEASURE6参照)

●ver.2.0式AD PCMデータ登録

n filename,PPC
n=filename,PPC
.Ook filename,PPC
.Ook=filename,PPC

・n:0～32767
・Ook:0-1C～09G
・filename:文字列
・PPC:後述

AD PCMデータの加工/登録を行う。

filenameで指定されたAD PCMファイルをトーン番号n、または.Ookで表すノートに登録する。PPC(PCM PROCESSING COMMAND)が指定してあればこれを実行し加工を施したAD PCMデータをnへ登録する。PPCは省略、あるいは複数設定可能。

なおnの設定範囲は0～32767となる。n(0～32767)による指定は.ADPCM_BANKの設定を無視して絶対的に行われる。

.Ookの指定はオクターブ値をo、音階kを文字列で表すMML形式で指定する。なお、oの設定範囲は-1～9、kはC,C#,D,D#,E,F,F#,G,G#,A,A#,Bのうちいずれか。

PPCはAD PCMデータを加工するコマンドで、次項のようなものがある。

filenameを、読み込むファイルの名前でなく、数値(0～32767)や音階MML(0-1C～09GまたはC-1～G9)を設定した場合は、ディスクからAD PCMデータを読み出すのではなく以前に同種のAD PCM加工/登録コマンドを用いて登録したトーンのコピーを用いて加工登録することになる。

登録先ノートやミキシングノートをOok書式で指定する場合、.ADPCM_BANKの設定によるバンク番号が考慮される。
(MEASURE6参照)

例

```
12 SNARE.PCM,P3,V150,M13,120,C0,1200,R,F-600,40
```

(SNARE.PCMの音程を3半音上に、音量を150%に増加させ、ディレイカウ

ント120でトーン13を合成、オフセット0カウントから1200カウント分を切り出し、これを反転再生に設定、オフセットカウント600まで音量40%からフェードインさせる。以上の加工処理を行ってトーン番号12へ登録する)
SNARE.PCMの部分をたとえば14に変更したとすると、

12 14,P3,V150,M13,120,C0,1200,R,F-600,40

加工登録対象を外部デバイスから読み込まず、登録済みのトーン番号14のデータを加工登録対象とする設定になる。

.ADPCM_BANK 3
.04C CYMBAL.PCM,V130,M03D

音量を150%上げたCYMBAL.PCMとバンク番号3のオクターブ3のDのトーンとを合成しこれをバンク番号3のオクターブ4のCのノートに登録する。

ver.2.0形式のPPC

・Pp
±1オクターブの範囲で音程変更を行う。変更幅は半音単位でpにて行う。pの設定範囲は-12〜+12。

・Vv
原音量を100とした音量の増減をvで指定する。vの設定範囲は0〜65535。数値が大きければ大きいほど音量が大きくなり、100より小さければ元の音量よりも小さくなる。

・Mm,d
トーン番号mのトーンと今回トーン番号nで登録するトーンとを合成する。その際、トーンnよりもPCMデータdカウント分遅らせて合成させることができる。mの設定範囲は0〜32767。dはトーンの先頭からのオフセット値で設定範囲は0〜4294967295。

・Cc,s
指定位置cから指定サイズsを切り出す。
cはトーンの先頭アドレスからのオフセット値で設定範囲は0〜4294967295。
sは切り出すサイズで設定範囲は0〜4294967295。

・R
再生方向を逆転させる処理を行う。

・Ff,l
フェードインまたはフェードアウトを行う。フェードインはオフセットを負値で与えることによって選択される。初期音量1から始まり、先頭からオフセット-fカウントの時点で原音量になるような音量変化になる。フェードアウトはオフセットを正値で与えることで選択される。初め、原音量から演奏が始まり、最後尾からオフセットfカウントの時点から徐々に音量が下がり始め最終的には音量1に落ち着く。
fの設定範囲は-2147483648〜2147483647。lの設定範囲は0〜127。lは0が無音、127が原音量より微妙に小さい音量に相当する。

●登録したAD PCMデータの削除

.ERASE n
.ERASE .0ok　　　(.0の'.'はあってもなくてもいい)

・n: 0〜32767
・0ok:0-1C〜09G

登録済みのAD PCMトーンのうち不要なものを削除する。

パラメータnや0okはAD PCM登録コマンドの同名パラメータと同じ書式。
0ok書式は.ADPCM_BANKの設定によるバンク番号が考慮される。
(MEASURE6参照)

例

.ADPCM_BANK 3
.ERASE 04C　　(バンク番号3のオクターブ4のCのトーンを削除する)

■外部ファイル読み込み

●外部ファイル読み込み

.CALL filename

・filename:文字列

filenameで指定したファイルを演奏時に読み込み実行する。
filenameで指定できるファイルはZPD,CNF,ZMD,ZMS(ZMC),MID,MDDで、拡張子の省略はできない。
後述の.INCLUDEとの違いはファイルの実行のされ方。.CALLでは演奏時に読み込んだファイル種別に応じて実行する。たとえばZPDファイルならばZPDを組み込みMIDならばMIDIポートへ出力する。一方、.INCLUDEは読み込んだファイルをZMSとしてその場でコンパイルしZMD化する。

例

.CALL CM-64TIMBRE.MID

●ZMSファイル読み込み

.INCLUDE filename

・filename:文字列

filenameで指定したファイルを読み込み、ZMSとしてコンパイルする。

filenameで指定できるファイルはZMS(ZMC),CNFで、拡張子省略時は'ZMC'が自動添付される。
前述の.CALLとの違いはファイルの実行のされ方。.CALLでは演奏時に読み込んだファイル種別に応じて実行する。たとえばZPDファイルならばZPDを組み込みMIDならばMIDIポートへ出力する。一方、.INCLUDEは読み込んだファイルをZMSとしてその場でコンパイルしてZMD化する。

例

.INCLUDE FMTONE.ZMS

■入出力MIDIポート選択

●カレントMIDI入力ポート選択

.CURRENT_MIDI_IN if

・if: 1〜4

MIDI入力を行う際のデフォルト・インタフェイスを選択する。MIDI入出力関連命令のほぼすべてに送受対象インタフェイスを指定できるようになっているが、この指定を省略した場合やver.2.0 ZMSコマンドなどの一部の送受対象インタフェイスを指定できない命令を使用したときに、本命令によって設定されたインタフェイスがデフォルトとして選択される。Z-MUSIC ver.3.0では.CURRENT_MIDI_INで設定したMIDIインタフェイスをカレントMIDI-INインタフェイスと呼ぶ。
ifの設定範囲は1〜4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。

例

.CURRENT_MIDI_IN 3
(MEASURE3 zm_current_midi_in()参照)

●カレントMIDI出力ポート選択

.CURRENT_MIDI_OUT if

・if: 1〜4

MIDI出力を行う際のデフォルト・インタフェイスを選択する。MIDI入出力関連命令のほぼすべてに送受対象インタフェイスを指定できるようになっているが、この指定を省略した場合やver.2.0 ZMSコマンドなどの一部の送受対象インタフェイスを指定できない命令を使用したときに、本命令によって設定されたインタフェイスがデフォルトとして選択される。Z-MUSIC ver.3.0では.CURRENT_MIDI_OUTで設定したMIDIインタフェイスをカレントMIDI-OUTインタフェイスと呼ぶ。
ifの設定範囲は1〜4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。

例

.CURRENT_MIDI_OUT 3
(MEASURE3 zm_current_midi_out()参照)

■MIDIデータ出力

●MIDIデータ出力

(Xn1,n2,…,ni)　　*

・n1〜ni: 0〜255

.current_midi_outで選択したMIDIポートへデータn1〜niを送信する。

データn1〜niの有効範囲は0〜255。256以上のデータも与えることができるが、その場合はその値を7ビットごとに分けて下位から送信する。たとえば8192は$2000だが、7ビットごとに分割するので$40,$00となる。これを下位から送信するので実際には$00,$40の順番でMIDI送信される。同様に32700は$3C,$7F,$01のように送信される。

例

(X $b0,$79,$00)
(MEASURE3 zm_midi_transmission()参照)

●MIDIデータ出力

.MIDI_DATA if,comment {n1,n2,…ni} ＊

・if:1～4

・comment:文字列

MIDIポートifへデータn1～niを送信する。データn1～niの有効範囲は0～255。256以上のデータも与えることができるがその場合はその値を7ビットごとに分けて下位から送信する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

commentは注釈で、'{'以外の255文字を用いて与えることができる。commentは省略可能。

例

```
.MIDI_DATA 2,PANPOT_CH1 {$b0,$0a,64}
```

(MEASURE3 zm_midi_transmission()参照)

●エクスクルーシブメッセージ送信

.EXCLUSIVE if,comment {n1,n2,…,ni} ＊

・if:1～4

・comment:文字列

MIDIポートifへデータn1～niをエクスクルーシブメッセージとしてMIDI送信する。データn1～niの有効範囲は0～127。128以上のデータも与えることができるが、その場合はその値を7ビットごとに分けて下位から送信する。

例

```
255→$7F,$01
8192→$00,$40
32700→$3C,$7F,$01
```

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

commentは注釈で、'{'以外の255文字を用いて与えることができる。commentは省略可能。

例

```
.EXCLUSIVE 2,GM_ON {$7e,$7f,$09,$01}
```

(MEASURE3 zm_exclusive()参照)

●ローランドエクスクルーシブメッセージ送信

.ROLAND_EXCLUSIVE dev,mdl,if,comment {n1,n2,…,ni} ＊

・dev:0～127

・mdl:0～127

・if:1～4

・comment:文字列

ローランド製のMIDI機器へエクスクルーシブメッセージを送信する。

エクスクルーシブヘッダやチェックサムバイトは送信メッセージ内容から自動算出し自動送信する。なお、送信の際、使用するローランドエクスクルーシブコマンドはDT1($12)。

・dev:デバイスID

同一機器間で区別をするためのID。通常は$10とすることが多い。

・mdl:モデルID

ローランドが同社製品に対して固有に与えた識別ID。たとえばGS音源は$42が与えられている。

・if:

送信を行うifを設定する。設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

・comment:コメント

'{'以外の255文字を用いて送信内容に関する注釈を与えることができる。省略可能。

・n1～ni:エクスクルーシブデータ

送信するエクスクルーシブデータを与える。有効範囲は0≦ni≦127。128以上は、その値を7ビットごとに分けて下位から送信する。

例

```
.ROLAND_EXCLUSIVE $10,$42,0,MASTERVOLUME=100 {$40,$00,$04$,$64}
```

(MEASURE3 zm_exclusive()参照)

●ヤマハエクスクルーシブメッセージ送信

.YAMAHA_EXCLUSIVE dev,mdl,if,comment {n1,n2,…,ni} ＊

・dev:0～127

・mdl:0～127

・if:1～4

・comment:文字列

ヤマハ製のMIDI機器へエクスクルーシブメッセージを送信する。

エクスクルーシブヘッダや転送バイト数、チェックサムバイトは送信メッセージ内容から自動算出し自動送信する。なお、送信の際、使用するヤマハエクスクルーシブタイプはバルクダンプ。

・dev:デバイスID

同一機器間で区別をするためのID。通常は$00とすることが多い。

・mdl:モデルID

ヤマハが同社製品に対して固有に与えた識別ID。たとえばXG音源は$4cが与えられている。

・if:

送信を行うifを設定する。設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

・comment:コメント

'{'以外の255文字を用いて送信内容に関する注釈を与えることができる。省略可能。

・n1～ni:エクスクルーシブデータ

送信するエクスクルーシブデータを与える。有効範囲は0≦ni≦127。128以上はその値を7ビットごとに分けて下位から送信する。

例

```
.YAMAHA_EXCLUSIVE $10,$4c,1,XG SYSTEM ON {$00,$00,$7E,$00}
```

(MEASURE3 zm_exclusive()参照)

●スタンダードMIDIファイルの送信

.MIDI_DUMP if,filename

.SMF if,filename

・if:1～4

・filename:文字列

ファイルネームfilenameのFORMAT0のスタンダードMIDIファイルをMIDIポートifに送信する。Z-MUSIC ver.2.0で扱われていたMIDIダンプデータ(MDD)も送信することもできる。

FORMAT1,FORMAT2のスタンダードMIDIファイルを送信することはできない。

filenameの拡張子省略時には(互換性保持のため)'.MDD'が自動添付される。スタンダードMIDIファイルの場合は'.MID'や'.SMF'といった拡張子を明記する必要がある。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

例

```
.MIDI_DUMP CM-64TIMBRE.MID
```

(MEASURE3 zm_transmit_midi_dump()参照)

■SC-55/mkII(SC-155/CM-300/CM-500)専用命令

●SC-55初期化

.SC55_INIT id,if

.SC55_RESET id,if

.GS_INIT id,if

.GS_RESET id,if

・id:$00～$7f

・if:1～4

MIDIポートifに接続されたSC-55の初期化を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-55のデバイスIDで省略可能、省略時は$10。

例

```
.SC55_INIT $10
```

●SC-55の各パートのボイスリザーブ

.SC55_V_RESERVE id,if {n1,n2,…,n16} ＊

.SC55_PARTIAL_RESERVE id,if {n1,n2,…,n16} ＊
.GS_V_RESERVE id,if {n1,n2,…,n16} ＊
.GS_PARTIAL_RESERVE id,if {n1,n2,…,n16} ＊
・id:$00～$7f
・n1～n16:0～127
・if:1～4
MIDIポートifに接続されたSC-55の各パートのボイスリザーブを行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n16)は必ず16個設定しなくてはならない。またパラメータ各値の総和は音源の同時発音数以内に収める必要がある。

n1～n16はそれぞれパート1～16のボイスリザーブ値に対応し、10番目のパラメータはリズムパートに対応する。

(SC-55マニュアルp79、またはMEASURE3 zm_gs_partial_reserve()参照)

例
```
        .SC55_PARTIAL_RESERVE $10 {
                2,4,2,2,0,0,0,0,0,5,0,0,0,0,0,0
        }
```

●SC-55のリバーブパラメータの設定
.SC55_REVERB id,if {n1,n2,…,n7} ＊
.GS_REVERB id,if {n1,n2,…,n7} ＊
・id:$00～$7f
・n1～n7:0～127
・if:1～4
MIDIポートifに接続されたSC-55のリバーブパラメータの設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n7)は7個まで任意の個数設定可能。
```
        n1:REVERB MACRO          (00:Room1,01:Room2,02:Room3,
                                  03:Hall1,4:Hall2,05:Plate,
                                  06:Delay,07:Panning Delay(FB))
        n2:REVERB CHARACTER             ($00～$07)
        n3:REVERB PRE-LPF               ($00～$07)
        n4:REVERB LEVEL                 ($00～$7F)
        n5:REVERB TIME                  ($00～$7F)
        n6:REVERB DELAY FEEDBACK        ($00～$7F)
        n7:REVERB SEND LEVEL TO CHORUS  ($00～$7F)
```
(SC-55マニュアルp79参照)

例
```
        .SC55_REVERB $10 ={4,4,0,88,70,0,0}
```

●SC-55のコーラスパラメータの設定
.SC55_CHORUS id,if {n1,n2,…,n8} ＊
.GS_CHORUS id,if {n1,n2,…,n8} ＊
・id:$00～$7f
・n1～n8:0～127
・if:1～4
MIDIポートifに接続されたSC-55のコーラスパラメータの設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n8)は8個まで任意の個数設定可能。
```
        n1:CHORUS MACRO          (00:Chorus1,01:Chorus2,02:Chorus3,
                                  03:Chorus4,04:FeedbackChorus,
                                  05:Flanger,06:ShortDelay0,
                                  07:ShortDelay(FB))
        n2:CHORUS PRE-LPF               ($00～$07)
        n3:CHORUS LEVEL                 ($00～$7F)
        n4:CHORUS FEEDBACK              ($00～$7F)
        n5:CHORUS DELAY                 ($00～$7F)
        n6:CHORUS RATE                  ($00～$7F)
        n7:CHORUS DEPTH                 ($00～$7F)
        n8:CHORUS SEND LEVEL TO REVERB  ($00～$7F)
```
(SC-55マニュアルp79参照)

例
```
        .SC55_CHORUS $10 ={2,0,100,8,80,3,19,0}
```

●SC-55のパートのパラメータを設定
.SC55_PART_SETUP pt,id,if {n1,n2,…,n119} ＊
.GS_PART_SETUP pt,id,if {n1,n2,…,n119} ＊
.SC55_PART_PARAMETER pt,id,if {n1,n2,…,n119} ＊
.GS_PART_PARAMETER pt,id,if {n1,n2,…,n119} ＊
・pt:1～16
・id:0～127
・if:1～4
・n1～n119:0～127
MIDIポートifに接続されたSC-55のパートのパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

ptはパートナンバーを表し、1≦pt≦16で省略は不可。またパートナンバー＝10はリズムパートに対応する。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n119)は119個まで任意の個数設定可能。
```
        n1:Rx CHANNEL              (1～16,17=OFF)
        n2:Rx PITCH BEND           (0～1:OFF/ON)
        n3:Rx CH PRESSURE(CAf)     (0～1:OFF/ON)
        n4:Rx PROGRAM CHANGE       (0～1:OFF/ON)
        n5:Rx CONTROL CHANGE       (0～1:OFF/ON)
        n6:Rx POLY PRESSURE(PAf)   (0～1:OFF/ON)
        n7:Rx NOTE MESSAGE         (0～1:OFF/ON)
        n8:Rx RPN                  (0～1:OFF/ON)
        n9:Rx NRPN                 (0～1:OFF/ON)
        n10:Rx MODURATION          (0～1:OFF/ON)
        n11:Rx VOLUME              (0～1:OFF/ON)
        n12:Rx PANPOT              (0～1:OFF/ON)
        n13:Rx EXPRESSION          (0～1:OFF/ON)
        n14:Rx HOLD1               (0～1:OFF/ON)
        n15:Rx PORTAMENTO          (0～1:OFF/ON)
        n16:Rx SOSTENUTO           (0～1:OFF/ON)
        n17:Rx SOFT                (0～1:OFF/ON)
        n18:MONO/POLY MODE         (0～1:MONO/POLY)
        n19:ASSIGN MODE     (0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
        n20:USE FOR RHYTHM PART          (0:OFF,1:MAP1,2:MAP2)
        n21:PITCH KEY SHIFT          ($28～$58:－24～＋24[semitone])
        n22:PITCH OFFSET FINE(upper)  ($08～$F8:－12.0～＋12.0[Hz])
        n23:PITCH OFFSET FINE(lower)           〃
        n24:PART LEVEL             (0～127)
        n25:VELOCITY SENSE DEPTH   (0～127)
        n26:VELOCITY SENSE OFFSET  (0～127)
        n27:PART PANPOT          (0:RANDOM,01(LEFT)～64～127(RIGHT))
        n28:KEY RANGE LOW          (0～127:C-1～G9)
        n29:KEY RANGE HIGH         (0～127:C-1～G9)
        n30:CC1 CONTROL NUMBER     (0～127)
        n31:CC2 CONTROL NUMBER     (0～127)
        n32:CHORUS SEND DEPTH      (0～127)
        n33:REVERB SEND DEPTH      (0～127)

        n34:Vibrato rate           ($0E～$72:－50～＋50)
        n35:Vibrato depth          ($0E～$72:－50～＋50)
        n36:TVF cutoff freq.       ($0E～$50:－50～＋16)
        n37:TVF resonance          ($0E～$72:－50～＋50)
        n38:TVF&TVA Env.attack     ($0E～$72:－50～＋50)
        n39:TVF&TVA Env.decay      ($0E～$72:－50～＋50)
        n40:TVF&TVA Env.release    ($0E～$72:－50～＋50)
        n41:Vibrato delay          ($0E～$72:－50～＋50)

        n42:SCALE TUNING C       ($00～$7F:－64～＋63[cent])
        n43:SCALE TUNING C#      ($00～$7F:－64～＋63[cent])
        n44:SCALE TUNING D       ($00～$7F:－64～＋63[cent])
```

```
n45:SCALE TUNING D#          ($00～$7F:－64～＋63[cent])
n46:SCALE TUNING E           ($00～$7F:－64～＋63[cent])
n47:SCALE TUNING F           ($00～$7F:－64～＋63[cent])
n48:SCALE TUNING F#          ($00～$7F:－64～＋63[cent])
n49:SCALE TUNING G           ($00～$7F:－64～＋63[cent])
n50:SCALE TUNING G#          ($00～$7F:－64～＋63[cent])
n51:SCALE TUNING A           ($00～$7F:－64～＋63[cent])
n52:SCALE TUNING A#          ($00～$7F:－64～＋63[cent])
n53:SCALE TUNING B           ($00～$7F:－64～＋63[cent])

n54:MOD PITCH CONTROL        ($28～$58:－24～＋24[semitone])
n55:MOD TVF CUTOFF CONTROL   ($00～$7F:－9600～＋9600[CENT])
n56:MOD AMPLITUDE CONTROL    ($00～$7F:－100.0～＋100.0[%])
n57:MOD LFO1 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n58:MOD LFO1 PITCH DEPTH     ($00～$7F:0～600[cent])
n59:MOD LFO1 TVF DEPTH       ($00～$7F:0～2400[cent])
n60:MOD LFO1 TVA DEPTH       ($00～$7F:0～100.0[%])
n61:MOD LFO2 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n62:MOD LFO2 PITCH DEPTH     ($00～$7F:0～600[cent])
n63:MOD LFO2 TVF DEPTH       ($00～$7F:0～2400[cent])
n64:MOD LFO2 TVA DEPTH       ($00～$7F:0～100.0[%])

n65:BEND PITCH CONTROL       ($28～$58:－24～＋24[semitone])
n66:BEND TVF CUTOFF CONTROL  ($00～$7F:－9600～＋9600[CENT])
n67:BEND AMPLITUDE CONTROL   ($00～$7F:－100.0～＋100.0[%])
n68:BEND LFO1 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n69:BEND LFO1 PITCH DEPTH    ($00～$7F:0～600[cent])
n70:BEND LFO1 TVF DEPTH      ($00～$7F:0～2400[cent])
n71:BEND LFO1 TVA DEPTH      ($00～$7F:0～100.0[%])
n72:BEND LFO2 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n73:BEND LFO2 PITCH DEPTH    ($00～$7F:0～600[cent])
n74:BEND LFO2 TVF DEPTH      ($00～$7F:0～2400[cent])
n75:BEND LFO2 TVA DEPTH      ($00～$7F:0～100.0[%])

n76:CAf PITCH CONTROL        ($28～$58:－24～＋24[semitone])
n77:CAf TVF CUTOFF CONTROL   ($00～$7F:－9600～＋9600[CENT])
n78:CAf AMPLITUDE CONTROL    ($00～$7F:－100.0～＋100.0[%])
n79:CAf LFO1 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n80:CAf LFO1 PITCH DEPTH     ($00～$7F:0～600[cent])
n81:CAf LFO1 TVF DEPTH       ($00～$7F:0～2400[cent])
n82:CAf LFO1 TVA DEPTH       ($00～$7F:0～100.0[%])
n83:CAf LFO2 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n84:CAf LFO2 PITCH DEPTH     ($00～$7F:0～600[cent])
n85:CAf LFO2 TVF DEPTH       ($00～$7F:0～2400[cent])
n86:CAf LFO2 TVA DEPTH       ($00～$7F:0～100.0[%])

n87:PAf PITCH CONTROL        ($28～$58:－24～＋24[semitone])
n88:PAf TVF CUTOFF CONTROL   ($00～$7F:－9600～＋9600[CENT])
n89:PAf AMPLITUDE CONTROL    ($00～$7F:－100.0～＋100.0[%])
n90:PAf LFO1 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n91:PAf LFO1 PITCH DEPTH     ($00～$7F:0～600[cent])
n92:PAf LFO1 TVF DEPTH       ($00～$7F:0～2400[cent])
n93:PAf LFO1 TVA DEPTH       ($00～$7F:0～100.0[%])
n94:PAf LFO2 RATE CONTROL    ($00～$7F:－10.0～＋10.0[Hz])
n95:PAf LFO2 PITCH DEPTH     ($00～$7F:0～600[cent])
n96:PAf LFO2 TVF DEPTH       ($00～$7F:0～2400[cent])
n97:PAf LFO2 TVA DEPTH       ($00～$7F:0～100.0[%])

n98:CC1 PITCH CONTROL        ($28～$58:－24～＋24[semitone])
n99:CC1 TVF CUTOFF CONTROL   ($00～$7F:－9600～＋9600[CENT])
n100:CC1 AMPLITUDE CONTROL   ($00～$7F:－100.0～＋100.0[%])
n101:CC1 LFO1 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n102:CC1 LFO1 PITCH DEPTH    ($00～$7F:0～600[cent])
n103:CC1 LFO1 TVF DEPTH      ($00～$7F:0～2400[cent])
n104:CC1 LFO1 TVA DEPTH      ($00～$7F:0～100.0[%])
n105:CC1 LFO2 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n106:CC1 LFO2 PITCH DEPTH    ($00～$7F:0～600[cent])
n107:CC1 LFO2 TVF DEPTH      ($00～$7F:0～2400[cent])
n108:CC1 LFO2 TVA DEPTH      ($00～$7F:0～100.0[%])
n109:CC2 PITCH CONTROL       ($28～$58:－24～＋24[semitone])
n110:CC2 TVF CUTOFF CONTROL  ($00～$7F:－9600～＋9600[CENT])
n111:CC2 AMPLITUDE CONTROL   ($00～$7F:－100.0～＋100.0[%])
n112:CC2 LFO1 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n113:CC2 LFO1 PITCH DEPTH    ($00～$7F:0～600[cent])
n114:CC2 LFO1 TVF DEPTH      ($00～$7F:0～2400[cent])
n115:CC2 LFO1 TVA DEPTH      ($00～$7F:0～100.0[%])
n116:CC2 LFO2 RATE CONTROL   ($00～$7F:－10.0～＋10.0[Hz])
n117:CC2 LFO2 PITCH DEPTH    ($00～$7F:0～600[cent])
n118:CC2 LFO2 TVF DEPTH      ($00～$7F:0～2400[cent])
n119:CC2 LFO2 TVA DEPTH      ($00～$7F:0～100.0[%])
```
(SC-55マニュアルp75,p79～80参照)

例

```
.SC55_PART_SETUP 16 {17,1,1,1}
```

●SC-55のドラムセットの設定

.SC55_DRUM_SETUP map,key,id,if {n1,n2,…,n8} *
.GS_DRUM_SETUP map,key,id,if {n1,n2,…,n8} *

- map:0,1
- key: 0～127
- id: 0～127
- if: 1～4
- n1～n8: 0～127

MIDIポートifに接続されたSC-55のドラムセットの設定を変える。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

mapはマップナンバーを表し設定範囲は0,1。

keyはノートナンバーを表し設定範囲は0≦key≦127。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n8)は8個まで任意の個数設定可能。

```
n1:PLAY KEY NUMBER      (0～127)
n2:LEVEL                (0～127)
n3:ASSIGN GROUP NUMBER  (0:NON,1～127)
n4:PANPOT               (0:RANDOM,01(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH         (0～127:0.0～1.0)
n6:CHORUS DEPTH         (0～127:0.0～1.0)
n7:Rx NOTE OFF          (0～1:OFF/ON)
n8:Rx NOTE ON           (0～1:OFF/ON)
```
(SC-55マニュアルp82参照)

例

```
.GS_DRUM_SETUP 0,24,$10 {24,127,0,0,10,15,0,1}
```

●SC-55のドラムセット名の設定

.SC55_DRUM_NAME map,id,if str
.GS_DRUM_NAME map,id,if str

- map:0,1
- id: 0～127
- if: 1～4
- str:文字列

MIDIポートifに接続されたSC-55のドラムセット名の設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

mapはマップナンバーを表し設定範囲は0,1。

idはSC-55のデバイスIDで省略可能、省略時は$10。

strはドラセット名を表す文字列で、"～"で括り12文字以内で指定する。

(SC-55マニュアルp82参照)

例

```
.GS_DRUM_NAME 0,$10 "USER SET"
```

●SC-55コンソールへの文字列表示

.SC55_PRINT id,if str
.GS_PRINT id,if str

- id: 0～127
- if: 1～4
- str:文字列

MIDIポートifに接続されたSC-55のコンソールに文字列を表示する。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

文字列strは"～"で括り32文字以内に収めなければならない。

idはSC-55のデバイスIDで省略可能、省略時は$10。

例

```
.GS_PRINT $10 "HELLO"
```

●SC-55グラフィックディスプレイへのドットパターン表示

.SC55_DISPLAY id,if {n1,n2,…,n16} ＊
.GS_DISPLAY id,if {n1,n2,…,n16} ＊

・id: 0～127
・if: 1～4
・n1～n16:%0000000000000000～%1111111111111111

MIDIポートifに接続されたSC-55のグラフィックディスプレイにドットパターン(16x16)を表示する。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n16)は必ず16個。

n1:1段目のドットパターン
n2:2段目のドットパターン
:
:
n16:16段目のドットパターン

例

```
.GS_DISPLAY $10 { %0001000000010000
                  %0000100000100000
                  %0111111111111100
                  %0000000100000000
                  %0011111111111000
                  %0000000100000000
                  %0111111111111100
                  %0001000100010000
                  %0000100100100000
                  %1111111111111110
                  %0000000000000000
                  %0011111111111000
                  %0010000000001000
                  %0010000000001000
                  %0011111111111000
                  %0010000000001000}
```

■GM音源専用命令

●GM音源初期化

.GM_SYSTEM_ON if

・if: 1～4

MIDIポートifに接続されたGM音源を初期化状態にリセットする。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

例

```
.GM_SYSTEM_ON
```

■SC-88/SC-88VL専用命令

●SC-88モード設定

.SC88_MODE_SET md,id,if

・md:0,1
・id: 0～127
・if: 1～4

MIDIポートifに接続されたSC-88のモードの設定を行う。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-88のデバイスIDで省略可能、省略時は$10。

mdはモード値を表し、md=0でSingleモード、md=1でDoubleモードが設定される。

(SC-88VLマニュアルp77参照)

例

```
.SC88_MODE_SET 1,$10
```

●SC-88リバーブパラメータの設定

.SC88_REVERB id,if {n1,n2,...,n8} ＊

・id: 0～127
・if: 1～4
・n1～n8: 下記参照

MIDIポートifに接続されたSC-88のリバーブパラメータの設定を行う。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-88のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n8)は8個以内で任意の個数設定可能。

```
n1:REVERB MACRO 00:Room1,01:Room2,02:Room3,03:Hall1,
                04:Hall2,05:Plate,06:Delay,07:Panning Delay(FB)
n2:REVERB CHARACTER           ($00～$07)
n3:REVERB PRE-LPF             ($00～$07)
n4:REVERB LEVEL               ($00～$7F)
n5:REVERB TIME                ($00～$7F)
n6:REVERB DELAY FEEDBACK      ($00～$7F)
n7:REVERB SEND LEVEL TO CHORUS ($00～$7F/SC-88ではダミー)
n8:REVERB PRE-DELAY TIME      ($00～$7F)
```

(SC-88VLマニュアルp77参照)

例

```
.SC88_REVERB $10 {0,0,127,10,10,0,127}
```

●SC-88コーラスパラメータの設定

.SC88_CHORUS id,if {n1,n2,...,n9}

・id: 0～127
・if: 1～4
・n1～n9: 下記参照

MIDIポートifに接続されたSC-88のコーラスパラメータの設定を行う。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-88のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n9)は9個以内で任意の個数設定可能。

```
n1:CHORUS MACRO 00:Chorus1,01:Chorus2,02:Chorus3,
                03:Chorus4,04:Feedback Chorus,
                05:Flanger,06:ShortDelay,07:ShortDelay(FB)
n2:CHORUS PRE-LPF              ($00～$07)
n3:CHORUS LEVEL                ($00～$7F)
n4:CHORUS FEEDBACK             ($00～$7F)
n5:CHORUS DELAY                ($00～$7F)
n6:CHORUS RATE                 ($00～$7F)
n7:CHORUS DEPTH                ($00～$7F)
n8:CHORUS SEND LEVEL TO REVERB ($00～$7F)
n9:CHORUS SEND LEVEL TO DELAY  ($00～$7F)
```

(SC-88VLマニュアルp78参照)

例

```
.SC88_CHORUS $10 {0,3,60,10,10,10,24,30,30}
```

●SC-88ディレイパラメータの設定

.SC88_DELAY id,if {n1,n2,...,n11} ＊

・id: 0～127
・if: 1～4
・n1～n11: 下記参照

MIDIポートifに接続されたSC-88のディレイパラメータの設定を行う。

ifの設定範囲は１～４。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはSC-88のデバイスIDで省略可能、省略時は$10。
バラメータ(n1〜n11)は11個以内で任意の個数設定可能。

```
n1:DELAY MACRO                 (00:Delay1,01:Delay2,
                                02:Delay3,03:Delay4,
                                04:Pan Delay1,
                                05:Pan Delay2,
                                06:Pan Delay3,
                                07:Pan Delay4,
                                08:Delay to Reverb,
                                09:Pan Repeat)
n2:DELAY PRE-LPF               ($00〜$07)
n3:DELAY TIME CENTER           ($01〜$73:0.1ms〜sec)
n4:DELAY TIME RATIO LEFT       ($01〜$78:4〜500%)
n5:DELAY TIME RATIO RIGHT      ($01〜$78:4〜500%)
n6:DELAY LEVEL CENTER          ($00〜$7F)
n7:DELAY LEVEL LEFT            ($00〜$7F)
n8:DELAY LEVEL RIGHT           ($00〜$7F)
n9:DELAY LEVEL                 ($00〜$7F)
n10:DELAY FEEDBACK             ($00〜$7F:-64〜+63)
n11:DELAY SENDLEVEL TO REVERB  ($00〜$7F)
```
(SC-88VLマニュアルp78参照)

例

```
.SC88_DELAY $10 {0,3,10,22,22,64,64,64,64,64,30}
```

●SC-88イコライザパラメータの設定

.SC88_EQUALIZER id,if {n1,n2,n3,n4} *

・id: 0〜127
・if: 1〜4
・n1〜n4:下記参照

MIDIポートifに接続されたSC-88のイコライザパラメータの設定を行う。

ifの設定範囲は1〜4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
idはSC-88のデバイスIDで省略可能、省略時は$10。
バラメータ(n1〜n4)は4個以内で任意の個数設定可能。

```
n1:EQ LOW FREQ                 (0,1:200Hz,400Hz)
n2:EQ LOW GAIN                 ($34〜$4c:-12〜+12dB)
n3:EQ HIGH FREQ                (0,1:3kHz,6kHz)
n4:EQ HIGH GAIN                ($34〜$4c:-12〜+12dB)
```
(SC-88VLマニュアルp78参照)

例

```
.SC88_EQUALIZER $10 {0,$40,1,$40}
```

●SC-88パートパラメータの設定

.SC88_PART_SETUP pt,id,if {n1,n2,...,n127} *
.SC88_PART_PARAMETER pt,id,if {n1,n2,...,n127} *

・pt:1〜16
・id: 0〜127
・if: 1〜4
・n1〜n127:下記参照

MIDIポートifに接続されたSC88パートパラメータの設定する。

ifの設定範囲は1〜4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
ptはパートナンバーを表し、1≦pt≦16で省略は不可。またパートナンバー=10はリズムパートに対応する。
idはSC-88のデバイスIDで省略可能、省略時は$10。
バラメータ(n1〜n127)は127個まで任意の個数設定可能。

```
n1:Rx CHANNEL                  (1〜16,17=OFF)
n2:Rx PITCH BEND               (0〜1:OFF/ON)
n3:Rx CH PRESSURE(CAf)         (0〜1:OFF/ON)
n4:Rx PROGRAM CHANGE           (0〜1:OFF/ON)
n5:Rx CONTROL CHANGE           (0〜1:OFF/ON)
n6:Rx POLY PRESSURE(PAf)       (0〜1:OFF/ON)
n7:Rx NOTE MESSAGE             (0〜1:OFF/ON)
n8:Rx RPN                      (0〜1:OFF/ON)
n9:Rx NRPN                     (0〜1:OFF/ON)
n10:Rx MODURATION              (0〜1:OFF/ON)
n11:Rx VOLUME                  (0〜1:OFF/ON)
n12:Rx PANPOT                  (0〜1:OFF/ON)
n13:Rx EXPRESSION              (0〜1:OFF/ON)
n14:Rx HOLD1                   (0〜1:OFF/ON)
n15:Rx PORTAMENTO              (0〜1:OFF/ON)
n16:Rx SOSTENUTO               (0〜1:OFF/ON)
n17:Rx SOFT                    (0〜1:OFF/ON)
n18:MONO/POLY MODE             (0〜1:MONO/POLY)
n19:ASSIGN MODE       (0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
n20:USE FOR RHYTHM PART        (0:OFF,1:MAP1,2:MAP2)
n21:PITCH KEY SHIFT         ($28〜$58:−24〜+24[semitone])
n22:PITCH OFFSET FINE(upper)  ($08〜$F8:−12.0〜+12.0[Hz])
n23:PITCH OFFSET FINE(lower)            //
n24:PART LEVEL                 (0〜127)
n25:VELOCITY SENSE DEPTH       (0〜127)
n26:VELOCITY SENSE OFFSET      (0〜127)
n27:PART PANPOT          (0:RANDOM,01(LEFT)〜64〜127(RIGHT))
n28:KEY RANGE LOW              (0〜127:C-1〜G9)
n29:KEY RANGE HIGH             (0〜127:C-1〜G9)
n30:CC1 CONTROL NUMBER         (0〜127)
n31:CC2 CONTROL NUMBER         (0〜127)
n32:CHORUS SEND LEVEL          (0〜127)
n33:REVERB SEND LEVEL          (0〜127)
n34:Rx BANK SELECT             (0:OFF,1:ON)
n35:Rx BANK SELECT LSB         (0:OFF,1:ON)

n36:PITCH FINE TUNE(Upper)($00 00〜$40 00〜$7f 7f:-100〜0〜+100)
n37:PITCH FINE TUNE(Lower)

n38:DELAY SEND LEVEL           (0〜127)

n39:Vibrato rate               ($00〜$7f:−64〜+63)
n40:Vibrato depth              ($00〜$7f:−64〜+63)
n41:TVF cutoff freq.           ($00〜$7f:−64〜+63)
n42:TVF resonance              ($00〜$7f:−64〜+63)
n43:TVF&TVA Env.attack         ($00〜$7f:−64〜+63)
n44:TVF&TVA Env.decay          ($00〜$7f:−64〜+63)
n45:TVF&TVA Env.release        ($00〜$7f:−64〜+63)
n46:Vibrato delay              ($00〜$7f:−64〜+63)

n47:SCALE TUNING C             ($00〜$7F:−64〜+63[cent])
n48:SCALE TUNING C#            ($00〜$7F:−64〜+63[cent])
n49:SCALE TUNING D             ($00〜$7F:−64〜+63[cent])
n50:SCALE TUNING D#            ($00〜$7F:−64〜+63[cent])
n51:SCALE TUNING E             ($00〜$7F:−64〜+63[cent])
n52:SCALE TUNING F             ($00〜$7F:−64〜+63[cent])
n53:SCALE TUNING F#            ($00〜$7F:−64〜+63[cent])
n54:SCALE TUNING G             ($00〜$7F:−64〜+63[cent])
n55:SCALE TUNING G#            ($00〜$7F:−64〜+63[cent])
n56:SCALE TUNING A             ($00〜$7F:−64〜+63[cent])
n57:SCALE TUNING A#            ($00〜$7F:−64〜+63[cent])
n58:SCALE TUNING B             ($00〜$7F:−64〜+63[cent])

n59:MOD PITCH CONTROL        ($28〜$58:−24〜+24[semitone])
n60:MOD TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n61:MOD AMPLITUDE CONTROL   ($00〜$7F:−100.0〜+100.0[%])
n62:MOD LFO1 RATE CONTROL    ($00〜$7F:−10.0〜+10.0[Hz])
n63:MOD LFO1 PITCH DEPTH       ($00〜$7F:0〜600[cent])
n64:MOD LFO1 TVF DEPTH         ($00〜$7F:0〜2400[cent])
n65:MOD LFO1 TVA DEPTH         ($00〜$7F:0〜100.0[%])
n66:MOD LFO2 RATE CONTROL    ($00〜$7F:−10.0〜+10.0[Hz])
n67:MOD LFO2 PITCH DEPTH       ($00〜$7F:0〜600[cent])
n68:MOD LFO2 TVF DEPTH         ($00〜$7F:0〜2400[cent])
n69:MOD LFO2 TVA DEPTH         ($00〜$7F:0〜100.0[%])

n70:BEND PITCH CONTROL       ($40〜$58:−24〜+24[semitone])
n71:BEND TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n72:BEND AMPLITUDE CONTROL  ($00〜$7F:−100.0〜+100.0[%])
n73:BEND LFO1 RATE CONTROL   ($00〜$7F:−10.0〜+10.0[Hz])
n74:BEND LFO1 PITCH DEPTH      ($00〜$7F:0〜600[cent])
```

n75:BEND LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n76:BEND LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n77:BEND LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n78:BEND LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n79:BEND LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n80:BEND LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n81:CAf PITCH CONTROL ($28～$58:－24～＋24[semitone])
n82:CAf TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n83:CAf AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n84:CAf LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n85:CAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n86:CAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n87:CAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n88:CAf LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n89:CAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n90:CAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n91:CAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n92:PAf PITCH CONTROL ($28～$58:－24～＋24[semitone])
n93:PAf TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n94:PAf AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n95:PAf LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n96:PAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n97:PAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n98:PAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n99:PAf LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n100:PAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n101:PAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n102:PAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n103:CC1 PITCH CONTROL ($28～$58:－24～＋24[semitone])
n104:CC1 TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n105:CC1 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n106:CC1 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n107:CC1 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n108:CC1 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n109:CC1 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n110:CC1 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n111:CC1 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n112:CC1 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n113:CC1 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n114:CC2 PITCH CONTROL ($28～$58:－24～＋24[semitone])
n115:CC2 TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n116:CC2 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n117:CC2 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n118:CC2 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n119:CC2 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n120:CC2 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n121:CC2 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n122:CC2 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n123:CC2 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n124:CC2 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n125:TONE MAP NUMBER (0:SELECTED,1:SC-55,2:SC-88)
n126:TONE MAP0 NUMBER (1:SC-55,2:SC-88)
n127:EQ ON/OFF (0:OFF,1:ON)
(SC-88VLマニュアルp79～p81参照)

例

.SC88_PART_SETUP 16 {17,1,1,1}

●SC-88のドラムセットアップ設定

.SC88_DRUM_SETUP map,key,id,if {n1,n2,...,n9} ＊

・map:0,1
・key: 0～127
・id: 0～127
・if: 1～4
・n1～n8: 0～127

MIDIポートifに接続されたSC-88のドラムセットアップの設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

mapはマップナンバーを表し設定範囲は0,1。

keyはノートナンバーを表し設定範囲は0≦key≦127。

idはSC-88のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n9)は9個まで任意の個数設定可能。

n1:PLAY KEY NUMBER (0～127)
n2:LEVEL (0～127)
n3:ASSIGN GROUP NUMBER (0:NON,1～127)
n4:PANPOT (0:RANDOM,0(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH (0～127:0.0～1.0)
n6:CHORUS DEPTH (0～127:0.0～1.0)
n7:Rx NOTE OFF (0～1:OFF/ON)
n8:Rx NOTE ON (0～1:OFF/ON)
n9:DELAY SEND LEVEL (0～127:0.0～1.0)
(SC-88VLマニュアルp82参照)

例

.SC88_DRUM_SETUP $10 {42,127,0,64,64,64,1,1,24}

●SC-88のドラムセット名の設定

.SC88_DRUM_NAME map,id,if str

・map:0,1　または USER65,USER66
・id: 0～127
・if: 1～4
・str:文字列

MIDIポートifに接続されたSC-88のドラムセット名の設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

mapはマップナンバーを表し設定範囲は0,1。ユーザードラムセットの名前を設定する場合にはUSER65,またはUSER66という文字列を与えて指定する。

idはSC-88のデバイスIDで省略可能、省略時は$10。

strはドラセット名を表す文字列で、"～"で括り12文字以内で指定する。(SC88マニュアルp82参照)

例

.SC88_DRUM_NAME USER66,$10 "USER SET"

●SC-88ユーザー音色の設定

.SC88_USER_INST bk,pgm,id,if {n1,n2,...,n11} ＊

・bk: 0～1
・pgm: 1～128
・id: 0～127
・if: 1～4
・n1～n11:下記参照

MIDIポートifに接続されたSC88ユーザー音色の設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

bkは設定先音色バンク番号を表し、設定範囲は0,1。bk=0がバリエーション番号64,bk=1がバリエーション番号65に対応する。

pgmは設定先音色番号で、設定範囲は1～128。

idはSC-88のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n11)は11個まで任意の個数設定可能。

n1:SOURCE TONE#(MAP) (1,2)
n2:BANK NUMBER MSB (0～127)
n3:PROGRAM NUMBER (1～128)
n4:Vibrato rate ($00～$7f:－64～＋63)
n5:Vibrato depth ($00～$7f:－64～＋63)
n6:TVF cutoff freq. ($00～$7f:－64～＋63)
n7:TVF resonance ($00～$7f:－64～＋63)
n8:TVF&TVA Env.attack ($00～$7f:－64～＋63)
n9:TVF&TVA Env.decay ($00～$7f:－64～＋63)
n10:TVF&TVA Env.release ($00～$7f:－64～＋63)
n11:Vibrato delay ($00～$7f:－64～＋63)
(SC-88VLマニュアルp82参照)

例

```
.SC88_USER_INST 0,4,$10 {1,0,10,64,64,64,64,64,64,64,64}
```

●SC-88のユーザードラムセットの設定

.SC88_USER_DRUM st,nt,id,if {n1,n2,...,n12}　＊

・st: 0～1
・nt: 0～127
・id: 0～127
・if: 1～4
・n1～n12: 下記参照

MIDIポートifに接続されたSC-88のユーザードラムセットの設定を行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

stは設定先ドラムセット番号で設定範囲は0,1。st=0がドラムセット番号65、st=1がドラムセット番号66に対応する。

ntは登録先ノート番号で設定範囲は0～127。

idはSC-88のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n12)は12個まで任意の個数設定可能。

```
n1:PLAY KEY NUMBER         (0～127)
n2:LEVEL                   (0～127)
n3:ASSIGN GROUP NUMBER     (0:NON,1～127)
n4:PANPOT                  (0:RANDOM,01(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH            (0～127:0.0～1.0)
n6:CHORUS DEPTH            (0～127:0.0～1.0)
n7:Rx NOTE OFF             (0～1:OFF/ON)
n8:Rx NOTE ON              (0～1:OFF/ON)
n9:DELAY SEND LEVEL        (0～127:0.0～1.0)
n10:SOURCE DRUM SET#(MAP)  (1～2)
n11:PROGRAM NUMBER         (0～127)
n12:SOURCE NOTE NUMBER     (0～127)
```

(SC-88VLマニュアルp82参照)

例

```
.SC88_USER_DRUM 0,32,$10 {24,127,0,64,10,10,0,1,10,1,10,24}
```

■MT-32/CM-32L/CM-64/CM-500専用命令

●MT-32の初期化

.MT32_INIT id,if
.MT32_RESET id,if

・id: 0～127
・if: 1～4

MIDIポートifに接続されたMT-32の初期化を行う

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはMT-32のデバイスIDで省略可能、省略時は$10。

例

```
.MT32_INIT $10
```

●MT-32の各パートのパーシャルリザーブ

.MT32_P_RESERVE id,if {n1,n2,…,n9}　＊
.MT32_PARTIAL_RESERVE id,if {n1,n2,…,n9}　＊

・id: 0～127
・if: 1～4
・n1～n9: 0～127

MIDIポートifに接続されたMT-32の各パートのパーシャルリザーブを行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはMT-32のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n9)は必ず9個なければならない。パラメータ各値の総和は32以内。

n1～n8はパート1～8のパーシャルリザーブ値に対応し、n9はリズムパートに対応する。

例

```
.MT32_PARTIAL_RESERVE $10 {2,2,2,2,0,0,1,2,4}
```

●MT-32リバーブパラメータの設定

.MT32_REVERB id,if {n1,n2,n3}　＊

・id: 0～127
・if: 1～4
・n1～n3: 0～127

MIDIポートifに接続されたMT-32のリバーブパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはMT-32のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n3)は3個まで任意の個数設定可能。

```
n1:REVERB MODE     (0:Room,1:Hall,2:Plate,3:Tap Delay)
n2:REVERB TIME     (0～7)
n3:REVERB LEVEL    (0～7)
```

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
.MT32_REVERB $10 {3,3,5}
```

●MT-32各パートのMIDIチャンネル設定

.MT32_PART_SETUP id,if {n1,n2,…,n9}　＊

・id: 0～127
・if: 1～4
・n1～n9: 0～127

MIDIポートifに接続されたMT-32の各パートのMIDIチャンネルを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはMT-32のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n9)は9個まで任意の個数設定可能。

n1～n8はパート1～8のMIDIチャンネル値に対応し、n9はリズムパートに対応する。

パラメータ(n1～n9)はMIDIチャンネル値なので1～16まで有効。ただし、0あるいは17以上を与えた場合はそのパートを無効(OFF)と見なす。

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
.MT32_PART_SETUP $10 {2,3,4,5,6,7,8,9,10}
```

●MT-32リズムキットの設定

.MT32_DRUM_SETUP n,id,if {n1,n2,n3,n4}　＊
.MT32_RHYTHM_SETUP n,id,if {n1,n2,n3,n4}　＊

・n: 24～87
・id: 0～127
・if: 1～4

MIDIポートifに接続されたMT-32のリズムキットの設定を変更する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

nは変更対象ノートナンバーを表し、設定範囲は24≦n≦87。

idはMT-32のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n4)は4個まで任意の個数設定可能。

```
n1:TIMBRE          (0～63:i1～64,64～127:r1～64)
n2:OUTPUT LEVEL    (0～100)
n3:PANPOT          (0～14)
n4:REVERB SWITCH   (0～1:OFF/ON)
```

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
.MT32_DRUM_SETUP 24,$10 {10,100,12,1}
```

●MT-32コモンパラメータの設定

.MT32_COMMON n,id,if {str,n1,n2,n3,n4}　＊

・n: 1～64
・id: 0～127
・if: 1～4
・str: 文字列
・n1～n4: 0～127

MIDIポートifに接続されたMT-32の音色のコモンパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
nは設定対象ティンバー番号で、設定範囲は1～64。
idはMT-32のデバイスIDで省略可能、省略時は$10。
strは音色名で、"～"で括り、10文字以内(省略不可)の文字列で指定する。
パラメータ(n1～n4)は4個まで任意の個数設定可能。

```
n1:Structure of Partial#1&2     (0～12:1～13)
n2:Structure of Partial#3&4     (0～12:1～13)
n3:PARTIAL MUTE                 (0～15)
n4:ENV MODE                     (0～1:NORMAL,NO SUSTAIN)
```

(MT-32マニュアルp34, CM-64マニュアルp29参照)
例

```
.MT32_COMMON 1,$10 {1,1,0,0}
```

●MT-32パーシャルパラメータの設定
.MT32_PARTIAL n,p,id,if {n1,n2,…,n58} *
・n: 1～64
・p: 1～4
・id: 0～127
・if: 1～4
・n1～n58: 0～127
MIDIポートifに接続されたMT-32の音色のパーシャルパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
nは設定対象ティンバー番号で設定範囲は1≦n≦64。
pは設定対象パーシャルナンバーで設定範囲は1≦p≦4。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n58)は58個まで任意の個数設定可能。

```
n1:WG PITCH COARSE          (0～96:C1,C#1,～,C9)
n2:WG PITCH FINE            (0～100:-50～+50)
n3:WG PITCH KEYFOLLOW       (0～16:-1,-1/2,-1/4,0,1/8,1/4,
                             3/8,1/2,5/8,3/4,7/8,1,
                             5/4,3/2,2,s1,s2)
n4:WG PITCH BENDER SW       (0～1:OFF/ON)
n5:WG WAVEFORM/PCM BANK     (0～3:SQU1,SAW1,SQU2,SAW2)
n6:WG PCM WAVE              (0～127:1～128)
n7:WG PULSE WIDTH           (0～100)
n8:WG PW VELO SENS          (0～14:-7～+7)
n9:P-ENV DEPTH              (0～10)
n10:P-ENV VELO SENS         (0～3)
n11:P-ENV TIME KEYF         (0～4)
n12:P-ENV TIME 1            (0～100)
n13:P-ENV TIME 2            (0～100)
n14:P-ENV TIME 3            (0～100)
n15:P-ENV TIME 4            (0～100)
n16:P-ENV LEVEL 0           (0～100:-50～+50)
n17:P-ENV LEVEL 1           (0～100:-50～+50))
n18:P-ENV LEVEL 2           (0～100:-50～+50)
n19:P-ENV SUSTAIN LEVEL     (0～100:-50～+50)
n20:END LEVEL               (0～100:-50～+50)
n21:P-LFO RATE              (0～100)
n22:P-LFO DEPTH             (0～100)
n23:P-LFO MOD SENS          (0～100)
n24:TVF CUTOFF FREQ         (0～100)
n25:TVF RESONANCE           (0～30)
n26:TVF KEYFOLLOW           (0～14:-1,-1/2,-1/4,0,1/8,1/4,
                             3/8,1/2,5/8,3/4,7/8,1,
                             5/4,3/2,2)
n27:TVF BIAS POINT/DIR      (0～127)
n28:TVF BIAS LEVEL          (0～14:-7～+7)
n29:TVF ENV DEPTH           (0～100)
n30:TVF ENV VELO SENS       (0～100)
n31:TVF ENV DEPTH KEYF      (0～4)
n32:TVF ENV TIME KEYF       (0～4)
n33:TVF ENV TIME 1          (0～100)
n34:TVF ENV TIME 2          (0～100)
n35:TVF ENV TIME 3          (0～100)
n36:TVF ENV TIME 4          (0～100)
n37:TVF ENV TIME 5          (0～100)
n38:TVF ENV LEVEL 1         (0～100)
n39:TVF ENV LEVEL 2         (0～100)
n40:TVF ENV LEVEL 3         (0～100)
n41:TVF ENV SUSTAIN LEVEL   (0～100)
n42:TVA LEVEL               (0～100)
n43:TVA VELO SENS           (0～100:-50～+50
n44:TVA BIAS POINT 1        (0～127)
n45:TVA BIAS LEVEL 1        (0～12:-12～0)
n46:TVA BIAS POINT 2        (0～127)
n47:TVA BIAS LEVEL 2        (0～12:-12～0)
n48:TVA ENV TIME KEYF       (0～4)
n49:TVA ENV TIME V_FOLLOW   (0～4)
n50:TVA ENV TIME 1          (0～100)
n51:TVA ENV TIME 2          (0～100)
n52:TVA ENV TIME 3          (0～100)
n53:TVA ENV TIME 4          (0～100)
n54:TVA ENV TIME 5          (0～100)
n55:TVA ENV LEVEL 1         (0～100)
n56:TVA ENV LEVEL 2         (0～100)
n57:TVA ENV LEVEL 3         (0～100)
n58:TVA ENV SUSTAIN LEVEL   (0～100)
```

(MT-32マニュアルp34, CM-64マニュアルp29参照)
例

```
.MT32_PARTIAL 1,1,$10 {64,50,11,1,1,127,50,7}
```

●MT-32パッチの設定
.MT32_PATCH n,id,if {n1,n2,…,n7} *
・n: 1～128
・id: 0～127
・if: 1～4
・n1～n7: 0～127
MIDIポートifに接続されたMT-32のパッチを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
nは設定対象パッチナンバーで、設定範囲は1≦n≦128。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n7)は7個まで任意の設定個数設定可能。

```
n1:TIMBRE GROUP      (0～3:a,b,i,r)
n2:TIMBRE NUMBER     (0～63)
n3:KEY SHIFT         (0～48:-24～+24)
n4:FINE TUNE         (0～100:-50～+50)
n5:BENDER RANGE      (0～24)
n6:ASSIGN MODE       (0～3:POLY1～4)
n7:REVERB SWITCH     (0～1:OFF/ON)
```

(MT-32マニュアルp35, CM-64マニュアルp30参照)
例

```
.MT32_PATCH 1,$10 {0,0,0,50,12,1,1}
```

●MT-32コンソール文字列表示
.MT32_PRINT id,if str
・id: 0～127
・if: 1～4
・str: 文字列
MIDIポートifに接続されたMT-32のコンソールに文字列を表示する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
idはMT-32のデバイスIDで省略可能、省略時は$10。
strは文字列で、"～"でくくり20文字以内で指定する。
(MEASURE3 zm_mt32_print()参照)
例

```
.MT32_PRINT $10 "HELLO!"
```

■U220/U20専用命令

●U220セットアップパラメータの設定

.U220_SETUP id,if {n1,n2,…,n7} ＊

・id: 0～127
・if: 1～4
・n1～n7: 0～127

MIDIポートifに接続されたU220のセットアップパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはU220のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n7)は必ず7個指定しなければならない。

n1:Chorus SW	(0～1:OFF/ON)
n2:Reverb SW	(0～1:OFF/ON)
n3:Rx Cntrl Channel	(1～17:1～16,OFF)
n4:Patch Change	(0～5:MAP1～MAP4,Dir,OFF)
n5:Timbre Change	(0～5:MAP1～MAP4,Dir,OFF)
n6:Rhythm Change	(0～5:MAP1～MAP4,Dir,OFF)
n7:R.Inst Assign	(0～5:MAP1～MAP4,Dir,OFF)

(U220マニュアルp46,p146,p148)

例

```
        .U220_SETUP $10 {1,1,1,5,4,4,4}
```

●U220パートパラメータの設定

.U220_PART_SETUP pt,id,if {n1,n2,…,n13} ＊

・pt: 1～6
・id: 0～127
・if: 1～4
・n1～n13: 0～127

MIDIポートifに接続されたU220のテンポラリパッチのパートパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

ptはパートナンバーを表し、設定範囲は1≦pt≦6。

idはU220のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n13)は必ず13個指定しなければならない。

n1:Timbre Number	(1～128)
n2:Voice Reserve	(0～30)
n3:Receive Channel	(1～17:1～16,OFF)
n4:Key Range Low	(0～127:C-1～G9)
n5:Key Range High	(0～127:C-1～G9)
n6:Velo Level	(0～1:Above/Below)
n7:Velo Threshold	(0～127)
n8:Output Assign	(0～4:Dry,Rev,Cho,Dir1,Dir2)
n9:Level	(0～127)
n10:Pan	(0～15:L7～M～R7)
n11:Rx Volume	(0～1:OFF/ON)
n12:Rx Pan	(0～1:OFF/ON)
n13:Rx Hold	(0～1:OFF/ON)

(U220マニュアルp58,p149参照)

例

```
        .U220_PART_SETUP 1,$10 {1,3,1,0,127,1,127,0,0,8,0,0,0}
```

●U220コモンパラメータの設定

.U220_COMMON id,if {n1,n2,…,n18}

・id: 0～127
・if: 1～4
・n1～n18: 0～127

MIDIポートifに接続されたU220のテンポラリパッチのコモンパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

idはU220のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n18)は必ず18個指定しなければならない。

n1:Chorus Type	(0～4:Chorus1,Chorus2,FB-Chorus,Flanger,Short Delay)
n2:Chorus Out Mode	(0～1:Pre Rev,Post Rev)
n3:Chorus Level	(0～31)
n4:Chorus Delay	(0～31)
n5:Chorus Rate	(0～31)
n6:Chorus Depth	(0～31)
n7:Chorus Feedback	(1～63:－31～0～＋31)
n8:Reverb Type	(0～7:Room1～3,Hall1～2,Gate Delay,CrossDelay)
n9:Reverb Time	(0～31)
n10:Reverb Level	(0～31)
n11:Reverb Delay Feedback	(0～31)
n12:Reverb Pre Delay Feedback	(0～31)
n13:Param1 #	(0～63:0～5,7～31,64～95,OFF)
n14:Param1 Param	(0:Timbre Level,1:Env Attack 2:Env Decay,3:Env Sustain 4:Env Release,5:A.Bend Depth 6:A.Bend Rate,7:Detune Depth 8:Vib Rate,9:Vib Wave Form 10:Vib Depth,11:Vib Delay 12:Vib Rise Time,13:Vib Mod Depth 14:Chorus Level,15:Chorus Rate 16:Chorus Feedback,17:Reverb Level 18:Delay Feedback)
n15:Param2 #	(Same as Param1 #)
n16:Param2 Param	(Same as Param1 Param)
n17:Param3 #	(Same as Param1 #)
n18:Param3 Param	(Same as Param1 Param)

(U220マニュアルp54,p149参照)

例

```
    .U220_COMMON $10 {0,1,20,20,20,20,32,0,2,2,2,2,63,0,63,0,63,0}
```

●U220音色パラメータ設定

.U220_TIMBRE n,id,if {str,n1,n2,…,n26} ＊

・n: 1～128
・id: 0～127
・if: 1～4
・str: 文字列
・n1～n26: 0～127

MIDIポートifに接続されたU220に音色パラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。

nは登録先音色番号を表し、設定範囲は1≦n≦128。

idはU220のデバイスIDで省略可能、省略時は$10。

strは音色名を表す文字列で、"～"で括り、12文字以内で指定する。

パラメータ(n1～n26)は必ず26個指定しなければならない。

n1:Tone Media	(0～31:I,1～31)
n2:Tone Number	(1～128)
n3:Timbre Level	(0～127)
n4:Velocity Sens	(1～15:－7～＋7)
n5:Channel Press Sens	(1～15:－7～＋7)
n6:Env Attack Rate	(1～15:－7～＋7)
n7:Env Decay Rate	(1～15:－7～＋7)
n8:Env Sustain Level	(1～15:－7～＋7)
n9:Env Release Rate	(1～15:－7～＋7)
n10:Pitch Shift Coarse	(8～56:－24～＋24)
n11:Pitch Shift Fine	(14～114:－50～＋50)
n12:Bend Range Lower	(0～15:－36,－24,－12～0)
n13:Bend Range Upper	(0～12)
n14:Channel After Sens	(0～27:－36,－24,－12～＋12)
n15:Poly After Sens	(0～27:－36,－24,－12～＋12)
n16:Auto Bend Depth	(0～27:－36,－24,－12～＋12)
n17:Auto Bend Rate	(0～15)
n18:Detune Depth	(0～15)
n19:Rate	(0～63)
n20:Waveform	(0～8)

n21:Depth (0～15)
n22:Delay (0～15)
n23:Rise Time (0～15)
n24:Modulation Depth (0～15)
n25:Ch After Sens (0～15)
n26:Poly After Sens (0～15)
(U220マニュアルp72,p149参照)
例

```
.U220_TIMBRE 1,$10    {"SYNTHE-2",
                       0,1,127,8,8,8,8,8,8,32,64,2,2,
                       14,14,14,0,0,0,0,0,0,0,0,0,0
                      }
```

●U220ドラムパラメータの設定

.U220_DRUM_SETUP id,if {n1,n2,…,n7} ＊

・id: 0～127
・if: 1～4
・n1～n7: 0～127

MIDIポートifに接続されたU220のテンポラリパッチのドラムパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n7)は必ず7個指定しなければならない。

n1:Rhythm Setup # (0～3:1～4)
n2:Voice Reserve (0～30)
n3:Receive Channel (1～17:1～16,OFF)
n4:Level (0～127)
n5:Level Boost Sw (0～1:OFF/ON)
n6:Rx Volume (0～1:OFF/ON)
n7:Rx Hold (0～1:OFF/ON)
(U220マニュアルp63,p149参照)

例

```
.U220_DRUM_SETUP $10 {0,5,10,127,0,1,0}
```

●U220ドラムセットノートパラメータの設定

.U220_DRUM_INST n,id,if {n1,n2,…,n20} ＊

・n:35-99
・id: 0～127
・if: 1～4
・n1～n20: 0～127

MIDIポートifに接続されたU220のテンポラリドラムセットの各ノートにおけるパラメータを設定する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
nは設定先のノート番号で、設定範囲は35≦n≦99。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n20)は20個以内任意の個数設定可能。

n1:Tone Media (0～31:I,1～31)
n2:Tone Number (1～128)
n3:Source Key (0～127:C－1～G9)
n4:Mute Inst (34～98:OFF,B1～D7)
n5:Inst Level (0～31)
n6:Velocity Sens (0～15)
n7:Env Mode (0～1:Sustain,No Sustain)
n8:Env Attack Rate (1～15:－7～＋7)
n9:Env Decay Rate (1～15:－7～＋7)
n10:Env Release Rate (1～15:－7～＋7)
n11:Pitch Shift Coarse (0～27:－36,－24,－12～＋12)
n12:Pitch Shift Fine (14～114:－50～＋50)
n13:Channel After Sens (0～27:－36,－24,－12～＋12)
n14:Poly After Sens (0～27:－36,－24,－12～＋12)
n15:Random (0～15)
n16:Auto Bend Depth (0～27:－36,－24,－12～＋12)
n17:Auto Bend Rate (0～15)
n18:Detune Depth (0～15)
n19:Output Assign (0～3:Dry,Rev,Cho,Dir1)
n20:Pan (0～15:L7～M～R7)
(U220マニュアルp72,p149参照)
例

```
.U220_DRUM_INST 42,$10 {0,1,24,34,31,
                        15,1,8,8,8,14,64,14,14,
                        0,14,0,0,0,8}
```

●U220ディスプレイ文字列表示

.U220_PRINT id,if str

・id: 0～127
・if: 1～4
・str:文字列

MIDIポートifに接続されたU220のディスプレイに文字列を表示する。具体的にはU220のテンポラリパッチの名前を設定することで行う。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
idはU220のデバイスIDで省略可能、省略時は$10。
strは表示文字列を表し、"～"で括り、12文字以内で指定する。
例

```
.U220_PRINT $10 "APPLE TREE"
```

■M1/M1EX/M1R/M1REX専用命令

●M1受信チャンネル設定

.M1_SETUP {ch1,ch2,…,ch8} ＊
.M1_MIDI_CH {ch1,ch2,…,ch8} ＊

・ch1～ch8: 1～16

M1のSEQ0の各パートの受信チャンネルを設定する。

パラメータ(ch1～ch8)は必ず8個指定しなければならない。設定範囲は1≦chi≦17で1～16はMIDIチャンネルの1～16を表し、17は「そのパートは受信しない」を意味する設定となる。

n1:パート1の受信MIDIチャンネル
n2:パート1の受信MIDIチャンネル
:
:
n8:パート1の受信MIDIチャンネル

例

```
.M1_SETUP {1,2,3,4,5,6,17,10}
```

●M1パートパラメータの設定

.M1_PART_SETUP {n1,n2,…,n40} ＊

・n1～n40:

M1のSEQ0の各パートのパラメータを設定する。

パラメータ(n1～n40)は必ず40個(5×8パート分)設定しなければならない。

PART 1
n1:PROGRAM NUMBER (0～199:I00～C99)
n2:OUTPUT LEVEL (0～99)
n3:KEY TRANSPOSE ($F4～$0C:－12～12)
n4:DETUNE ($CE～$32:－50～＋50)
n5:PAN (0～13:10:0～0:10,C,C+D,D)
PART 2～8(n6～n40)以下同様の書式構成
(M1マニュアルp126参照)
例

```
.M1_PART_SETUP { 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
                 0,99,0,0,5
               }
```

●M1エフェクトパラメータの設定

.M1_EFFECT_SETUP {n1,n2,…,n25} ＊

・n1～n25:

　M1のSEQ0のエフェクトパラメータの設定を行う。パラメータ(n1～n25)は必ず25個指定しなければならない。

```
n1:Effect 1 Pattern No.        (0～33:1～32,Thru)
n2:Effect 2 Pattern No.        (0～33:1～32,Thru)
n3:Effect 1 L-Ch E.Balnc       (0～100)
n4:Effect 1 R-Ch E.Balnc       (0～100)
n5:Effect 2 L-Ch E.Balnc       (0～100)
n6:Effect 2 R-Ch E.Balnc       (0～100)
n7:Output 3 Pan                (0～101:OFF,100:0～0:100)
n8:Output 4 Pan                (0～101:OFF,100:0～0:100)
n9:Effect I/O        (BIT0:Effect 1 L-Ch 0=OFF/1=ON
                      BIT1:Effect 1 R-Ch 0=OFF/1=ON
                      BIT2:Effect 2 L-Ch 0=OFF/1=ON
                      BIT3:Effect 2 R-Ch 0=OFF/1=ON)
                      BIT4:Effect 2 0=PARALLEL/1=SERIAL)
n10～n17:Effect 1 Parameter
n18～n25:Effect 2 Parameter
```

Structure of Effect Parameter

(ofs=10 or 18)

1～3:Hall (4,5:Room,6:Live Stage)

```
n(ofs+0):Reverb Time      (0～97:0.2～9.9),(0～48:0.2～5.0)
n(ofs+1):Dummy             0
n(ofs+2):High Damp         (0～99)
n(ofs+3):Pre Delay         (0～200)
n(ofs+4):E/R Level         (0～99)
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

7～9:Early Reflection 1,2,3

```
n(ofs+0):E/R Time          (0～70:100～800)
n(ofs+1):Pre Delay         (0～200)
n(ofs+2):Dummy             0
      :                    :
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

10:Stereo Delayt,11:Cross Delay

```
n(ofs+0):Delay Time L (L)  (0～500)
n(ofs+1):Delay Time L (H)
n(ofs+2):Feedback          ($9D～$63:－99～99)
n(ofs+3):High Damp         (0～99)
n(ofs+4):Delay Time R (L)  (0～500)
n(ofs+5):Delay Time R (H)
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

12～13:Stereo Chorus 1～2 (14～15:Flanger)

```
n(ofs+0):Depth             (0～99)
n(ofs+1):Speed          (0～99:0.03～3.00 0.03step
                         100～199:3.1～13.0 0.1step
                         200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status     (BIT0:Waveform =0:Sin,=1:Tri
                         BIT1:Phase =0:0.0°,=1:180°
                         BIT2:Wave Shape =0:Normal
                                         =1:for Flanger)
n(ofs+3):Feedback          ($9D～$63:－99～＋99)
n(ofs+4):Delay Time        (0～200),(0～50)
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

16～17:Phase Shifter 1～2

```
n(ofs+0):Depth             (0～99)
n(ofs+1):Speed          (0～99:0.03～3.00 0.03step
                         100～199:3.1～13.0 0.1step
                         200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status     (BIT0:Waveform =0:Sin,=1:Tri
                         BIT1:Phase =0:0.0°,=1:180°
                         BIT2:Wave Shape =0:Normal
                                         =1:for Flanger)
n(ofs+3):Feedback          ($9D～$63:－99～＋99)
n(ofs+4):Manual            (0～99)
n(ofs+5):Dummy             0
n(ofs+6):Dummy             0
n(ofs+7):Dummy             0
```

18～19:Stereo Toremolo 1～2

```
n(ofs+0):Depth             (0～99)
n(ofs+1):Speed          (0～99:0.03～3.00 0.03step
                         100～199:3.1～13.0 0.1step
                         200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status     (BIT0:Waveform =0:Sin,=1:Tri
                         BIT1:Phase =0:0.0°,=1:180°
                         BIT2:Wave Shape =0:Normal
                                         =1:for Flanger)
n(ofs+3):Shape             ($9D～$63:－99～＋99)
n(ofs+4):Dummy             0
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

20:Equalizer

```
n(ofs+0):Dummy             0
      :                    :
n(ofs+3):Dummy             0
n(ofs+4):Low fc            (0～2:0.25k,0.50k,1.00k)
n(ofs+5):High fc           (0～2:1k,2k,4k)
n(ofs+6):High Gain         ($F4～$0C:－12～＋12)
n(ofs+7):Low Gain          ($F4～$0C:－12～＋12)
```

21:Overdrive

```
n(ofs+0):Dummy             0
n(ofs+1):Dummy             0
n(ofs+2):Drive             (0～99)
n(ofs+3):Level             (0～99)
n(ofs+4):Dummy             0
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

22:Distortion

```
n(ofs+0):Dummy             0
n(ofs+1):Dummy             0
n(ofs+2):Distortion        (0～99)
n(ofs+3):Level             (0～99)
n(ofs+4):Dummy             0
n(ofs+5):Dummy             0
n(ofs+6):Dummy             0
n(ofs+7):EQ Low Gain       ($F4～$0C:－12～＋12)
```

23:Exciter

```
n(ofs+0):Blend             ($9D～$63:－99～＋99)
n(ofs+1):Emphatic Point    (0～9:1～10)
n(ofs+2):Dummy             0
      :                    :
n(ofs+5):Dummy             0
n(ofs+6):EQ High           ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low            ($F4～$0C:－12～＋12)
```

24:Synphonic Ensamble

```
n(ofs+0):Depth             (0～99)
n(ofs+1):Dummy             0
      :                    :
n(ofs+5):Dummy             0
```

n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

25:Roteffect Speaker
n(ofs+0):Depth (0～99)
n(ofs+1):Dummy 0
n(ofs+2):Speed Rate ($F6～$0A:－10～＋10)
n(ofs+3):Dummy 0
:
n(ofs+7):Dummy 0

26:Delay/Hall
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Reverb Time (0～97:0.2～9.9)
n(ofs+5):Dummy 0
n(ofs+6):High Damp (0～99)
n(ofs+7):Pre Delay (0～150)

27:Delay/Room
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Reverb Time (0～97:0.2～9.9)
n(ofs+5):Dummy 0
n(ofs+6):High Damp (0～99)
n(ofs+7):Pre Delay (0～150)

28:Delay/Early Reflection
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):E/R Time (0～30:100～400)
n(ofs+5):Pre Delay (0～150)
n(ofs+6):Dummy 0
n(ofs+7):Dummy 0

29:Delay/Chorus
n(ofs+0):Delay Time L (L) (0～500)
n(ofs+1):Delay Time L (H)
n(ofs+2):Feedback L ($9D～$63:－99～99)
n(ofs+3):High Damp L (0～99)
n(ofs+4):Delay Time R (L) (0～500)
n(ofs+5):Delay Time R (H)
n(ofs+6):Feedback R ($9D～$63:－99～99)
n(ofs+7):High Damp R (0～99)

30:Delay/Chorus (31:Delay Flanger)
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
n(ofs+6):LFO Status (BIT0:Waveform ＝0:Sin,＝1:Tri
BIT1:Phase ＝0:0.0°,＝1:180°
BIT2:Wave Shape ＝0:Normal
＝1:for Flanger)
n(ofs+7):Feedback 0,($9D～$63:－99～99)

32:Delay/Phaser
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step)
n(ofs+6):Feedback ($9D～$63:－99～99)
n(ofs+7):Dummy 0

33:Delay/Tremolo
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step)
n(ofs+6):Dummy 0
n(ofs+7):Shape ($9D～$63:－99～99)
(M1マニュアルp127参照)
例
```
.M1_EFFECT_SETUP {1,3,50,50,50,50,50,50,%11111,
                  50,0,50,100,50,0,0,0,50,0,50,
                  100,50,0,0,0
                 }
```

●M1ディスプレイ文字列表示
.M1_PRINT str
・str:文字列
M1のディスプレイに文字列を表示する。具体的にはM1のSEQ0のソングネームの設定をすることで行う。

strは表示文字列で、"～"で括り、10文字以内で指定する。
例
```
.M1_PRINT "MY WAY"
```

●M1へ設定を送信する
.SEND_TO_M1 id,if
・id: 0～127
・if: 1～4
.M1_SETUP
.M1_PART_SEUP
.M1_EFFECT_SETUP
.M1_PRINT
で設定したパラメータ群をMIDIポートifに接続されたM1へ送信する。

ifの設定範囲は1～4。MIDIインタフェイスの個数が少ない場合は4より小さくなる。ifを省略した場合は.current_midi_outで選択したカレントMIDIポートへ出力される。
idはM1のデバイスIDで省略可能、省略時は$30。
なお、M1のデバイスIDは$30＋グローバルチャンネル(0～$f)で求められる(グローバルチャンネルはGLOBALモードのF5－1で設定できる)。
必ず、
.M1_SETUP
.M1_PART_SEUP
.M1_EFFECT_SETUP
.M1_PRINT
を設定してからこの命令を実行すること。'.M1_EFFECT_SETUP'のみ省略が可能。省略時はZ-MUSICが持つデフォルト設定がM1へ送信され設定されることになる。
.M1_SETUP
.M1_PART_SEUP
.M1_EFFECT_SETUP
.M1_PRINT
の間にM1関係以外のコマンドがあると正常なデータが送信されないので注意すること。
例
```
.SEND_TO_M1 $30
```

■波形メモリ

●波形メモリの登録
.WAVE_FORM wv,lm,ls,le,lt,comment {d0,d1,d2,…,d4294967295} *

.WAVEFORM wv,lm,ls,le,lt,comment {d0,d1,d2,…,d4294967295} ＊

・wv:8-255
・lm:０～２　または　1SHOT,REPEAT,ALTERNATE
・ls:０～4294967295
・le:０～4294967295
・lt:０～4294967295
・comment:文字列
・d0,d1,d2,…,d4294967295
　波形メモリを登録する。

　定義した波形はピッチモジュレーション(MML @M)、ARCC(MML @A)、エンハンスドベロシティシーケンス(MML [VELOCITY.DEPTH])、アゴーギク(MML [AGOGIK.DEPTH])にて利用可能。波形は整数で表し、各要素は離散的な波形構成値と見なされる。
・wv:定義先波形番号
　定義先波形番号を指定する。0～7はプリセット波形用に予約されているためユーザー定義波形である波形メモリは8～255に定義することになる。
・lm:ループモード
　後述するループ領域を度のようにループさせるかを設定する。なおループモード0～2と実際の波形処理のされ方の対応は以下の通り。0～2はそれぞれ1SHOT,REPEAT,ALTERNATEといった文字列で指定することも可能。省略時はlm=0 1SHOTが選択される。

　　lm=0 1SHOT　　→
　　　波形を一度実行したら最後の値を継続する
　　lm=1 REPEAT　　→→→→....
　　　ループ終点に到達したらループ開始点に戻る
　　lm=2 ALTERNATE →←→←....
　　　ループ開始点からループ終点まで交互に反復する

・ls:ループ開始点
　ループ開始位置を波形データの何番目にするかを設定する。設定範囲は０～4294967295。省略時は0が選択される。
・le:ループ終了点
　ループ終了位置を波形データの何番目にするかを設定する。設定範囲は０～4294967295。省略時は最後の波形データの位置が自動設定される。
・lt:ループ回数
　ループ領域を何回ループさせるかを設定する。設定範囲は０～4294967295。なお0は無限ループとして設定される。
・comment:コメント
　'{'以外の255文字を用いて波形に対する注釈を与えることができる。
・d1～d4294967295:波形データ
　波形データは符号付き16ビット整数－32768～＋32767で構成する。
　最大4294967296個。
　波形データは波形を利用する機能の種別に関係無く直値が利用される。値が小さすぎるときや大きすぎるときは、その効果が現れなかったり大きすぎたりするため、そのような場合には、振幅制御の命令を駆使するなどして適宜調整する必要がある。

例

```
.wave_form 8,REPEAT,800,1200,0,ORG_WAVE001 {0,5,-5,10,3,-300,10,6,80,10}
```

(MEASURE3 zm_set_wave_form1(),zm_set_wave_form2()参照)

●マクロ定義

.DEFINE macro p1,p2,...,p255 { contents } ＊

・macro:文字列
・p1～p255:文字列
・contents:文字列
　マクロ定義を行う。

　Z-MUSICには初めから備わっていないオリジナルのZMSコマンドを作成したり、頻繁に登場する演奏パターンをコマンド化してZMSを見やすくしたりすることができる。ただし、定義したマクロの数に応じてコンパイル処理は多少遅くなる(ただし、コンパイルしてできたZMDは通常のZMDと同様の処理速度で演奏される)。
　マクロ名はmacroで与え、'{'を除く英数字、全角の240文字以内の文字列で与えることができる。
　p1～p255はマクロに与えるパラメータで、%でマクロに与えたいパラメータの個数だけ、記述する。
　contentsはマクロの内容で、任意の文字列を与えることができる。p1～p255で設定したパラメータを内容中に挿入したいときには%を添付する。

例

```
.define DRUM1 %,%,%,% {
      c%d%c%d%
      }
```

　なお、.DEFINE命令には以下のような制限事項がある。
1)　マクロ名に全角文字は全文字使えるが、半角文字の場合は英字の大文字小文字の区別はない。
2)　ZMS中のありとあらゆるテキストがマクロ変換対象になる。
3)　マクロの変換対象となるソースは親ソースのみで、includeしたサブファイルはマクロの変換対象にならない。
4)　マクロ内容にパラメータに相当しない%記号を用いられない。
　たとえば、

```
.define ABC %,%,% {c%d%e%001}
```

としたとき最後の%001は2進数値と見なされずにマクロパラメータ識別子%と10進数の'001'として見なされる。
5)　.DEFINE文で与えたパラメータの数とマクロ内容で参照するパラメータの数は必ず同一でなければならない。たとえば、

```
.define ABC  %,%,% {c%d%e}
```

はエラーとなる。
　ただし、使用するときは与えるパラメータが少なくても構わない。足りないパラメータについてはなにも与えないで処理される。たとえば

```
.define ABC  %,%,% {c%d%e%}
```

のような定義の下では、

```
ABC 1,2
```

は、

```
c1d2e
```

と変換される

6)　%の後ろに数値をつけてパラメータの対応をわかりやすく記述することもできるが、その場合数値指定なしの記述とは混在できない。

例

○

```
.define DRUM1 %1,%2 {
      c%2d%1
      }
```

×

```
.define DRUM1 %,%2 {
      c%d%2
      }
```

■コメント/注釈文

●コメント

/str

・str:文字列

　'/'以後、改行までをコメントとして無視する。
　コンパイル後のZMDにはその内容は出力されない。

例

```
/ ORGAN SOLO
```

●コメント

.COMMENT str

・str:文字列
　改行までの文字列strをコメントとして無視する。

　'/'とは、コンパイル後のZMDにその内容が出力される点が違う。
　曲のタイトル、制作日時、データ制作者の名前などをZMSファイル先頭に記述するのが一般的な用途。

タイトル文字は、
・1行目は曲のタイトル。それ以降はコメント。
・曲名/作曲者/編曲者/データ作成者のような一般的な情報を複数行にわたって記述する場合は、以下の文字列で表される情報タイトルの後に記載すること。
・情報タイトル一覧

曲名:	TITLE:
作曲者:	COMPOSER:
編曲者:	ARRANGER:

作成者:　　　MANIPULATER:
作成日:　　　DATE:　　　　(原曲の作曲年月日はCOPYRIGHT:)
録音場所　　　STUDIO:
使用機器:　　INSTRUMENTS:
ジャンル:　　CATEGORY:
(情報タイトルの日本語は全角、英字は半角英数大文字のみ、また : は必須)

例)

```
.comment {
TITLE: I am a boy, therefore I have balls.
作曲者:西川善司
STUDIO:自宅
ジャンル:童謡
}
```

●文字列表示

.PRINT str

・str:文字列

改行までの文字列strを画面に出力する。
どんな方法で演奏しても文字列を画面に出力してしまうので、通常は用いない。

例

```
.PRINT HERE WE GO!
```

●ダミー命令

.DUMMY

なにも実行しないダミー命令。
ただし、ZMDに1バイト分のダミーコードが出力される。コンパイル後のZMDに手を加えるときなどに有効。通常は使用する必要はない。

例

```
.DUMMY
```

MEASURE 5

MMLコマンド

ここではZ-MUSICのMMLについて解説します。

5．1． MMLの文法について

Z-MUSICは基本的にはシャープ/ハドソン製のOPMDRV.Xに上位コンパチである。つまりOPMDRV.Xに存在したMMLはほぼ共通に使用することができ、OPMDRV.X用のデータをほとんど変更なしで演奏が可能だ。

Z-MUSICではMMLの小文字、大文字の区別を一切していない。よって状況、ユーザー各位の趣味に応じて自由に使い分けが行える(本節では説明の都合上、MMLを大文字で、パラメータを小文字で記述している)。

省略してもよいパラメータはその旨を記述しているが、それ以外の場合は省略できない。また、数値パラメータは頭に'$'をつければ16進数が、'%'をつければ２進数を設定することができる。なにもつけなければ10進数と見なされる。

各コマンドの頭についている印は各音源の種類を表している。すべてのコマンドがすべての音源に対応しているわけではないので注意が必要だ。対応していないデバイスに対してそのコマンドを用いると無視される。

[FM] 内蔵FM音源パート
[ADPCM] 内蔵AD PCMパート
[MIDI] MIDI音源パート

5．2． 基本MML

ここではZ-MUSIC ver.2.0以前から存在するMMLコマンドの仕様について解説する。ver.3.0になって機能が拡張されたものも存在するので注意が必要。

OPMDRV.Xとの互換性について

Z-MUSIC．Ｘでは初代のOPMDRV.Xと互換性がありますが以下の点に注意してください（OPMDRV3.Xとは互換性はありません）。

1． レジスタ番号48～55(KF:キーフラクション)を操作するＹコマンドは処理の都合上、MML @K相当に変換してしまうため、そのトラックにアサインされたチャンネル以外のキーフラクションを操作していた場合は動作がOPMDRV.Xとは異なってきます。

2． OPMDRV.Xは、'm_play()'や'm_cont()', 'm_stop()'などの演奏制御命令における操作対象パートをチャンネル番号で指定するような仕様でした。Z-MUSICではトラック番号で指定するように変更されています。

■音符、休符

[FM] [ADPCM] [MIDI] **音符**

As,l,g,v
Bs,l,g,v
Cs,l,g,v
Ds,l,g,v
Es,l,g,v
Fs,l,g,v
Gs,l,g,v

・s:#,+,-,!
・l:1-32767
・g:1-32767
・v:0-127

ＡＢＣ～Ｇはハ長調のラシド～ソに対応する。

・s:#，＋，－，！

変化記号。#，＋をつければ半音上がり、－をつけると半音下がる。！はナチュラルを表し、後述の[K.SIGN]([KEY_SIGNATURE])で調号設定を行った場合でも、調号を無効にした音高でその音符を演奏する。

・l:1～32767

lは音長を表し、1が全音符、2が２分音符、4が４分音符...となる。

・g:1～32767

ゲートタイム。実際の消音までの時間。lと同じく1が全音符、2が２分音符、4が４分音符...となる。

・v:0～127

ベロシティ。127がもっとも音圧が大きく、１がもっとも小さい。０は消音と見なされる。

lはマスタークロック設定コマンド'.MASTER_CLOCK'によって設定範囲が変動する。具体的には、

1≦l≦マスタークロックの値

が設定範囲となる。以下にデフォルトのマスタークロック＝192のときの音楽的音長と絶対音長との対応を示す。

音楽的音長	絶対音長	音楽的音長	絶対音長
1	192	12	16
2	96	16	12
3	64	24	8
4	48	32	6
6	32	64	3
8	24	192	1

絶対音長＝int(マスタークロック/音楽的音長)

パラメータs,l,g,vはいずれも省略可能。

ｌが省略されている場合は後述のL/@Lで設定したデフォルト音長が採択される。

gが省略されている場合はlの値をもとに後述のQ/@Qで指定したゲートタイムパラメータをもとにゲートタイムが計算される。

vが省略された場合は後述のU/@Uで指定したデフォルトベロシティが採択される。

ｌおよびgの前に'*'をつけると絶対音長指定になる。絶対音長とはZ-MUSICが処理する音長カウンタのことでデフォルトでは全音符が192になっている。このとき、たとえばC4とC*48は同時間発音される。'*'の後ろには0～32767までの数値を指定することができる。

例１

a#4,2(ステップタイムを４分音符長で、ゲートタイムが2分音符長、ベロシティはデフォルトが採択され、A#が演奏される)

b*386,,125 （ステップ=386,ベロシティ=125でBが演奏される）

音長の後ろに'.'を設定すると付点音符と見なされる。

例２

g4. （付点4分音符）

e4..,*24.(ステップ=二重付点4分音符,ゲート=絶対音長36(=24+24*0.5)

共通コマンド.LENGTH_MODEで「ステップタイムモード」とした場合は'*'がついている/ついていないに関係なく絶対音長指定となる。

FM ADPCM MIDI　休符

Rl,g

・l:1〜32767

・g:1〜32767

休符。

l,gの意味については音符と同じ。省略時は音符と同じようにデフォルトパラメータが起用される。

例

r8

r*96

休符におけるゲートタイムgは通常の音符から休符へ後述の'&'によるタイ指定が行われたときのみ意味をなす。たとえば、

c4&r4,8

としたときはc4は４分音符音長分発音し続け、休符に差し掛かる。休符の長さは４分音符音長だがゲートタイムは８分音符に設定されているので前の音の c の音は休符が始まってから８分音符音長後に消音されることとなる。

共通コマンド.LENGTH_MODEで「ステップタイムモード」とした場合には'*'がついている、ついていないに関係なく絶対音長指定となる。

FM ADPCM MIDI　ウェイト

@Wl

・l:1〜32767

指定時間、前の状態を保存する。lは音長を表す。音符や休符と同じく与えた値は音楽的音長として認識されるが'*'を添付して値を与えた場合は絶対音長として処理される。l省略時はデフォルトが起用される。

例

@w*64

@w4..

休符にはゲートタイムがあるが、@Wにはないのが特徴。よって、たとえば、

c4&@w4

とした場合、@w4が終了してもcの音は鳴り続ける。

■調号

FM ADPCM MIDI　調号

[K.SIGN k1,k2,…,kn]

・k1〜kn:文字列

調号を設定する。どの音階にどの変化記号（#，♭）がつくのかを設定する。kn部分は変化記号、音階の順に設定し、１個ずつカンマ（ ，）で区切る。

例

[K.SIGN +c,+d,+f,+g]（ホ長調）
cdfgに自動的に#(シャープ)がつく

[K.SIGN -a,-b,-d,-e]（変イ長調）
abdeに自動的に♭(フラット)がつく

[K.SIGN -a,+b]
aに♭(フラット)が、bに#(シャープ)が自動的につく

また以下の調名を記述することによって、一般的な調号の設定を行うこともできる。

Cmajor,Gmajor,Dmajor,Amajor,Emajor,Bmajor,F+major,F#major,
C+major,C#major,Fmajor,B-major,Bbmajor,E-major,Ebmajor,
A-major,Abmajor,D-major,Dbmajor,G-major,Gbmajor,C-major,
Cbmajor,Aminor,Eminor,Bminor,F+minor,F#minor,C+minor,C#minor,
G+minor,G#minor,D+minor,D#minor,A+minor,A#minor,Dminor,Gminor,
Cminor,Fminor,B-minor,Bbminor,E-minor,Ebminor,A-minor,Abminor

例

[K.SIGN G-Major]　　変ト長調(cdegabに♭)

ナチュラル記号である'!'を使用すれば一時的に調号の効力をなくすことができる。また、調号の効力をなす音階に対してさらに#,+(#)や-(♭)がついた場合は調号と両方考慮した音階になる。

例

[K.SIGN G-Major] b!

（この調ではbには通常ならば♭がつき、b-になる(半音下がる)のだが！のためナチュラル指定となり、bで演奏される）

[K.SIGN G-Major] b-

（この調ではbには通常ならば♭がつき、b-になる(半音下がる)のだが♭のためb--(=a)で 演奏される）

■オクターブ

FM ADPCM MIDI　オクターブ

On

・n:-1〜9

オクターブを設定する。初期値は4。nは－1〜9まで設定可能。

実際の発音可能域は各デバイスによって異なる。

・FM音源

オクターブ０のＤ＃からオクターブ８のＤまで。

・AD PCM

登録しなければ発音はしないが、セレクトバンク内の(@コマンド、MEASURE6参照)オクターブ－１のＣからオクターブ９のＧまで対応。

・MIDI

オクターブ－１のＣからオクターブ９のＧまで。実際は各楽器によって違う。詳しくは各MIDI楽器のマニュアルのインプリメンテーションチャートなどを参照のこと。

FM ADPCM MIDI　相対オクターブ

<

>

オクターブを相対的に変更する。<はオクターブを＋1し、>はオクターブを－1する。設定範囲を超えた場合はエラーとなる。

■音長制御

FM ADPCM MIDI　デフォルト音長設定

Ln

・n:1〜32767

@Ln

・n:1〜32767

デフォルト音長を設定する。音長をパラメータに持つMMLにおいて音長が省略された場合、この命令で設定した値がパラメータとして与えられる。

Ｌのパラメータnは音楽的音長、絶対音長のいずれかで書くことができる。

音楽的音長の場合はn分音符といった意味となる(設定範囲については音符MMLの説明を参照)。絶対音長による指定は音名のときと同じように数値の前に'*'を記述して設定できる。設定範囲は0≦n≦32767。

@Lのパラメータnは絶対音長のみで設定範囲は0≦n≦32767。

初期値はL4(=@L48)である。

L,@Lいずれの場合も音長数値の後ろに付点が設定可能。

例

l16.

@l96

l*2000.

FM ADPCM MIDI　ゲートタイム設定

Qn

・n:-8〜8

１音中で実際に発音している時間を音長のn/8時間単位で設定する。

初期値 n=8、設定範囲は-8〜8。値が正の場合、１が発音時間がもっとも短く、8がもっとも長い(テヌート)。

値が負の場合はステップタイムよりもゲートタイムを長くすることができる。たとえばq-8とするとゲートタイムはステップタイムの２倍に設定され、q-4ならばゲートタイムはステップタイムの1.5倍に設定される。

分母の値はデフォルトでは8だが共通コマンド.GATETIME_RESOLUTIONにて8,16,32,64,128のいずれかへ変更が可能。たとえば.GATETIME_RESOLUTION 128と設定された場合、MML Qには-128〜128の値を与えることができるようになる。

例

q4 cdefg　　　　（スタッカート気味になる）

q-4 cdefg　　　（前の音がこれから鳴る音と重なるようになる）

@Qを設定したときはQは機能せず、逆にQを設定したときは@Qは機能しない。あとに設定したものが有効となる。

FM ADPCM MIDI　ゲートタイム設定
@Qn
・n:-32767～32767

ノートオフ（キーオフ,発音をやめること）を絶対音長でnカウント早める。nの値が発音音長より大きい値の場合、@Qは機能せずQ8の状態で演奏される。たとえば、

@Q96 C*16

の場合は発音長が 16－96<0 となるので@Qの処理はされず発音長は16カウントとなる。

nを負値に設定した場合は絶対音長で-nカウント、ノートオフを遅らせることができる。たとえば、

@Q-12 c*48

とするとcのゲートタイムは60(=12+48)となる。

初期値は@Q0に相当し設定範囲は-32767～32767。

@Qを設定したときはQは機能せず、逆にQを設定したときは@Qは機能しない。あとに設定したものが有効となる。

■タイ/スラー

FM ADPCM MIDI　タイ/スラー
&

前後の音をつなぐ。同音高の場合は全デバイスにおいてタイとなる。

例

a4&a16.

異音名の場合は、各デバイスによって対応が異なる。

・FM音源パート

スラーとなる

例

a4&a#4　　（滑らかにaからa#へつながる）

・AD PCMパート

後述のPCMモード([PCM_MODE]コマンドにより設定)によって対応が異なる。

通常モード時([PCM_MODE RHYTHM])

'&'を除いたときのように処理される

例

a4&a#4 → a4a#4のように演奏される

楽器PCMモード時([PCM_MODE TIMBRE])

例

a4&a#4　　（滑らかにaからa#へつながる）

・MIDI

後述のタイモード(@Jコマンドにより設定)によって対応が異なる。

通常モード時(@J0)

'&'を除いたときのように処理される

例

a4&a#4 → a4a#4のように演奏される

FM音源部互換モード時(@J1)

スラーとなる。

例

a4&a#4(滑らかにaからa#へつながる)

FM ADPCM MIDI　加算式タイ
^n
・n:1～

音符の直後に書くことによって、nで指定した音長分をその音符の音長に加算する。nの後ろに付点を書けば、付点を考慮した音長が加算される。nには前に'*'をつけて絶対音長を記述することもできる。

例

c4^16
d^*1000.
'ceg4^16'
(d2^8,e)

MIDI　MIDIパートタイモード
@Jn
・n:0,1

MIDI部のタイの処理モードを設定する。

n=0で通常モード(初期値)、n=1でFM音源部互換モードになる。

c&d

とした場合、通常(@J0時)はMIDIでは&のない、

c d

として処理される。

一方@J1時ではFM音源部と同様に、ある音の発音後、その音の持続音(減衰音)で次の音の音程へ変化するようになる。よって、FM音源部での、

(c<d)&(d>c)

といった表現や、後のコラムで「動作の相違がある」と書かれている、

(g,e)24&e
@b0,683 c4&c+4

絶対音長0

Z-MUSICでは休符、ウェイト、和音とポルタメントを除いて、絶対音長0も使用可能になっています(音楽的音長0は使用不可)。音長がゼロの音符は発音処理を行うだけです。操作対象デバイスによって多少の効果の相違がありますので注意してください。

●MIDI

c*0e*0g*0

ではcegの和音が鳴りっぱなしになります。

MIDI楽器のでリズムパートを持つものは、たいてい、ノートオンのメッセージのみで演奏可能です(ノートオフの処理を省略しても演奏に影響しない)。

こういったリズムパートをシーケンスする場合に絶対音長0は有用です。

たとえば02C＝バスドラム，02D＝スネアドラム，03C+＝クラッシュシンバルのような ローランド系リズムキットで、

L4 o3c+*0 |:4 o2 cdcd :|

とすると、最初の小節の頭だけシンバルが鳴り、２回目以降は鳴らないといったシーケンスができます。

絶対音長０の音符は楽器の最大同時発声数内で１トラックにつきいくつでも設定可能です。

絶対音長０の音符はそのままでは鳴りっぱなし状態になりますが、これを('&'をつけて)タイで通常音符へつなぐとその通常音符が鳴り終わった時点で絶対音長０の音もノートオフすることができます。

c*0&e*0&g4

とするとcegの和音が４分音符長で鳴ります。

この'&'で結ぶ書式では最大16和音までが記述できます。16和音のときは絶対音長0の音符が15個と通常音長の音符がひとつという構成になります。絶対音長0の音符を15個以上記述した場合は古い順にノートオフされない音が出てきます。

専用和音コマンドによる和音との混在は可能であるため、状況状況に応じて使い分けができます。また、この2種類の和音書式間のタイ/スラーも可能です。

●FM音源/ADPCM音源

基本的にモノフォニックですので、

c*0e*0g*0

のような表記で和音を発音させることはできません。ただし、[VOICE_RESERVE]命令でそのトラック内で複数の発音を宣言した場合には、その範囲内で複数音の発音が可能になります。その場合は、MIDIパートと同じように、

c*0&e*0&g4

のような記述で和音の発音が可能です。専用和音コマンドによる和音との混在もでき、この２つの書式間のタイ/スラーも可能です。

AD PCM音源の場合はMPCM.Xが組み込まれている場合にのみ,以上の動作が保証されます。組み込まれていないときは完全に単声仕様(モノフォニック)になります。

のような表記もFM音源部と同様に機能するようになる(ただし、MIDIの仕様制限のため、最初に発音した音高から１オクターブを超えることはできない)。

■連符

FM ADPCM MIDI　連符

{～} n

・n:1～32767

{～} で囲まれたMML群を合計の音長がn分音符音長になるように演奏する。nには付点をつけたり、'*'をつけて絶対音長による指定も行える。

{～} の中には音長を操作するMMLを書くことはできない（付点, L, @ L など）。

n省略時はデフォルト音長が起用される。

例

```
{cdef}2
{abc}4..
{cd&de}*192
```

■和音

FM ADPCM MIDI　和音

'MML…'n,dly(&)

・n:1-32767

・dly:1-32767

「'」(シングルコーテーション)に囲まれた音符を同一タイミングで発音する。'～'間に書けるMMLは以下に限られる。

音符	c～b(16個まで)
調号	# + - !
音長	1～(付点も可)
絶対音長	*1～*32767
ｵｸﾀｰﾌﾞｺﾏﾝﾄﾞ	0-1～09
ｵｸﾀｰﾌﾞｽｲｯﾁ	< >

一度に16和音まで発音が可能。17音以上音階を書くとエラーとなる。

音長は'～'内で最後に設定されたものが和音全体の音長として採用される。

オクターブコマンド、オクターブスイッチは'～'内だけ有効。

'～'内に音長を書かずに外に数値ｎを書くと、これは'*'をつけるつけないに変わらず絶対音長の指定になる。この場合の設定範囲は0≦n≦32767で全音符以上の発音時間も設定できる。

ディレイパラメータdlyを設定することによって、和音を一度に発音せず、各音を絶対音長dlyずつ時間的にずらして１音ずつ発声する(アルペジオ奏法)。ディレイは初期値0で、省略可能だが省略時は以前設定した値がそのまま起用される。

例

```
'c4eg'
```

ドミソを４分音符で演奏する。

```
o4'gb<d'96
```

ソシレを絶対音長96で演奏する。和音発音後のデフォルトオクターブは4に変わりないことに注意。

```
'c4eg<c',6
```

まずcが発音され、その6カウント後にeが発音され、さらに6カウント後ずつg, <cが発音される。

ディレイ値は必ず全部の音が発声可能な値でなければエラーとなる。たとえば、

```
'c4eg',30
```

などはgが発音できないのでエラーとなる。

同構成音の和音～和音間、異構成音の和音～和音間、単音～和音間、和音～単音間を'&'で結んでのタイ/スラーが可能。ただし処理の関係上、タイでつながれた和音のディレイは強制的に0に設定される。つまり、

```
'c4eg',6&'e4gb-',6
```

は、

```
'c4eg',6&'e4gb-',0
```

として処理される。

また、オートベンドやモジュレーション、アフタータッチシーケンス、ARCCなど、すべての特殊コマンドが和音においても有効である。もちろん同時にこれらを和音に対して使用できる。

FM音源部での和音を用いた場合、チャンネルが競合すると正常に発音されないので注意すること。たとえば、

```
.assign TR1,FM1
.assign TR2,FM2
.assign TR3,FM3
.track1 'ceg'48
.track2 b-           ←正常に演奏されない
.track3 f+           ←正常に演奏されない
```

のような場合、.track1から発音する'ceg'48の和音はFM音源チャンネル1, 2,3が使用されるが、.track2と.track3がこのときそれぞれ b-とf+をチャンネル2,3で演奏しているためにチャンネルの競合が発生している。この場合、トラック1の和音を正常に発音させるには、

```
.assign TR1,FM1
.assign TR2,FM4
.assign TR3,FM5
.track1 'ceg'48
.track2 b-           ←正常に演奏されない
.track3 f+           ←正常に演奏されない
```

のようにしてチャンネル2,3をあけるようにしなければならない。

ver.3.0では和音の1音1音に対して独立した発音ベロシティを設定可能になった。

例

```
'c,,88e,,99,g,,127'
```

(cegをそれぞれベロシティ88,99,127で発音する。)

■音量/ベロシティ

FM ADPCM MIDI　音量

Vn

・n:0-16

音量の設定を行う。設定範囲は0≦n≦16。初期値は8。数値が大きいほど音量が大きくなる。AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

```
V11 cde
```

FM ADPCM MIDI　音量

@Vn

・n:0-127

音量の設定を細かく行う(絶対音量と呼ぶ)。設定範囲は0≦n≦127。初期値はMIDI,AD PCMで64、FM音源では106。数値が大きいほど音量が大きくなる。AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

```
@v100 c
```

VとＶ

FM音源とAD PCM(MPCM組み込み時)、MIDI音源では@VとVの対応関係が異なります。これはFM音源の出力音量が指数関数的に増減するのに対してMIDIとAD PCM音源では線形に増減するからです。以下に音源別にに@ＶとＶの対応表を示します。

FM音源

Vn	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
@Vn	85	87	90	93	95	98	101	103	106	109	111	114	117	119	122	125	127

MIDI音源,AD PCM音源

Vn	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
@Vn	0	8	16	24	32	40	48	56	64	72	80	88	96	104	112	120	127

以上のことから相対ボリューム_n ‾nの使用結果においても結果が異なります。たとえば、

```
v8 _1
```

とした場合FM音源パートでは結果が@V105(=106-1)に相当しますが、MIDIパートでは@V63(=64-1)になります。

FM ADPCM MIDI　相対音量

‾n

_n

・n:0-127

音量の相対指定を行う。‾は音量増加を表し、_は音量減少を表す。設定

範囲は0≦n≦127で、演算の結果、オーバーフローを起こした場合は０または127へ修正される。また計算は@Vn表される絶対音量レベルで行われる。
nを省略した場合は以前に設定したものが起用される。nの初期値は１。
AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

@v120 _10 c _ d ¯ e　(cは@120－10＝@v110で、dは@v110－10＝@v100で、eは@v100＋10＝@v110で演奏される)

|:10 _10 c :|　(だんだんと音量が下がりながらcが演奏される)

FM ADPCM MIDI　ベロシティ

@Un

Un

・n:0-127

ベロシティの設定を行う。ベロシティとは鍵盤を押す強さ、すなわち音量に相当する。

MIDI楽器によってはこれによって音のニュアンスを変えることができる。設定範囲は0≦n≦127。数値が大きいほど音量が大きくなる。

FM音源パート,AD PCM音源パートでは@Vと同じに扱われる。

初期値はMIDIパートのみ127で、FM音源,ADPCM音源では@Vの初期値と同じ。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

FM ADPCM MIDI　相対ベロシティ

@U±n

U±n

・n:-127～+127

ベロシティの相対指定を行う。設定範囲は－127≦n≦＋127。演算によってオーバーフローを起こした場合の音量値は０または127に修正される。

初期値は＋1。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

@u127c @u-10d @u+5e

u127c　u-10d　u+5e

(c, d, eはそれぞれベロシティ127, 117, 122で発音する)

FM ADPCM MIDI　相対ベロシティ再設定

@U

以前設定した@U相対ベロシティを再び実行する。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

@u127 c @u-10 d @u e

(最後の@uもまた@u-10で機能するため e は@U107(=127-10-10)で発音する)

Uと@U

機能的にはまったく同じですが、演奏時にリアルタイムに考慮されるか否かの相違があります。

@u100|:4 @u+5 c :|

としたとき４回とも@u105でcが演奏されますが

@u100|:4 u+5 c :|

としたときは@u105, @u110, @u115, @u120でそれぞれ演奏されます。

なお、管理は@uとuはまったく別です。ですから、

@u5　|:3 @u+50 c u-5:| d

としたとき@u55で3回cが演奏され@u50でdが演奏されます。

uのみの指定は、それまでに最後に設定した@uをもう１回設定するので、

@u110|:10 u-2 c :|u e

のように、ループで変化してしまったベロシティを元に戻すことができます。この例では、直前のベロシティ値によらずeは@u110で演奏されます。

ただし、

@u110 c @u-2 d u e

のように、@U相対ベロシティを使用した場合は@Uの値自体が変化してしまうので、Uのみの設定は意味を持たなくなります。この例ならば e は d と同じく@U108(110-2)で演奏されます。

このようにU,@Uを使い分ければ、複雑で高度なシーケンスをすることができます。

@uは、ループなどの影響を受けない大域的な設定、uはループ等の影響を受ける局所的な設定というわけです。

FM ADPCM MIDI　ベロシティ再設定

U

以前設定した@Uの値を再設定する。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

例

@u127 c u-10 d u e　(c,d,eがそれぞれベロシティ127, 117, 127で発音する)

ベロシティのはたらきについて

FM音源パートとADPCM音源パートではベロシティと音量がまったく同じ機能として取り扱われます。たとえば、FM音源パートで、

V8　@U106　@V106

は同じ機能を果たします。

MIDI音源パートでは発音音量はベロシティと音量の2つの関係で決定されます。

ベロシティと音量を、ともに最大値を設定したとき、最大音量となります。

FM ADPCM MIDI　ベロシティシーケンス

Zn1,…,n128

・ni:0～127

ベロシティのばらつきを設定する。これをZ-MUSICではベロシティシーケンスと呼ぶ。各値の設定範囲は0≦ni≦127。

これは最大128個の音符MML、和音MML、ポルタメントMMLのベロシティを先に設定するものである。

たとえば、

z127,99,80　　cdefg

とするとcはベロシティ127で、dはベロシティ99で、eはベロシティ80で、ここでfは再びベロシティ127で、gはベロシティ99で発音される。

値は最大128個まで記述することができ、1個のみ指定したときは@Uとまったく同様に機能する。数値の頭に±を記述すれば相対指定が可能である。

数値をまったく書かずに'Z'のみ記述するとベロシティシーケンスのOFFと解釈される。また、ベロシティコマンド@U/Uを使用してもベロシティシーケンスは解除される。

ベロシティシーケンスはループコマンドを無視して処理されるので、

z99,100,127 |:100 c :| d e

とした場合cが100回ベロシティ99で、ループ外のdがベロシティ100で、eがベロシティ127で演奏される。

FM音源では音量とベロシティが同義に扱われるので、設定された値に対応した絶対音量で発音されることになる。

MIDIパートでは、楽器によってはベロシティの変化にともなって音色のニュアンスや音色を切り変えることができるのでそういった機能と合わせて使えばかなり興味深い効果を得ることができる。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

■キートランスポーズ/ディチューン/オートベンド

FM ADPCM MIDI　キートランスポーズ

Kn

・n:-127～127

音程を半音単位で上下させる（キートランスポーズ）。nはn半音を意味し、設定範囲は-127≦n≦127(±10オクターブ程度に相当)。初期値は0、またニュートラル（中央値）も0である。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

FM ADPCM MIDI　ディチューン

@Kn

・n:-7680～+7680

キートランスポーズを1/64半音単位で行う(いわゆるディチューン)。設定可能範囲は-7680≦n≦7680(±10オクターブ程度に相当)。初期値は０、またニュートラル(中央値)も0である。

Kと@Kはまったく管理が別である。たとえば、

@k10 cdefg

の頭に k1 をつけて、

k1 @k10 cdefg

としたとすると1半音と10/64半音上にピッチシフトされてcdefgが演奏される。

いちいちMMLを書き直すことなく各パートのMMLの頭にKコマンドを書くだけでディチューン等も考慮した転調が実現できる。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

FM ADPCM MIDI ディチューン
@Bn

・n:-8192～+8191

ベンド値(ディチューン)を設定する。@Kとまったく同じ機能をすると考えてよいが1オクターブ≒8192で換算される点が違う。ユーザーがMIDI音源に慣れている場合は@Bを、FM音源に慣れている場合は@Kを使うとよい。設定範囲は-8192≦n≦8191で、初期値は0、ニュートラル(中央値)も0である。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

FM ADPCM MIDI オートベンド
@Kn1,n2,dly,bnd

・n1:-7680～+7680　　開始ディチューン値(64:半音)
・n2:-7680～+7680　　終端ディチューン値(64:半音)
・dly:0～32767　　遅延時間
・bnd:-32767～+32767　　(正)変化所用時間,(負)継続時間,(0)いずれも設定しない

@Kの後ろに2つ以上のパラメータを書くとオートベンドの設定となる。オートベンドとはディチューンを一定時間内に滑らかに変化させる表現方法のこと。

n1はベンド開始ディチューン値

n2はベンド終了ディチューン値、

dlyはディレイ値でベンドの開始までの遅延時間に相当する。単位は絶対音長。

bndは正値の場合は変化にかける時間を表す。負値の場合では、ベンドを完了してどのくらいの時間、ベンド終了ディチューンを継続するかを設定する。0にするとディレイ以後、発音する音符の音長のすべてを使ってベンド終了ディチューンまで変化する設定となる。

dlyとbndは省略可能。省略時は以前の設定値が起用される。

初期値はn1,n2,dly,bndともに0である。

オートベンドを実行中に@Kおよび@Bにてディチューンを設定した場合、オートベンドの方は以降、強制的にOFFとされる。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

オートベンド中も各音符はQコマンドの影響を受ける点にも注意すること。たとえば、

@k0,768,0 q4 b-

とした場合は1オクターブ上のb-に達する前にキーオフしてしまう。

例) (ベンドレンジ(*1)が1オクターブになっている場合)

@k0,-64,24 c1

発音後、24カウント後から音程が下がり始めキーオフまでに半音(64=1半音)下がる。

@k0,128,12,24 c*64

では発音後12カウント後音程が上がり始め、ここから24カウントで目的の音高にたどり着き、28(64-12-24)カウント分、+128の音高を保ち続ける。

(*1)ベンドレンジはFMは±1オクターブに固定、MIDIは後述の@Gコマンドで変更可能である。

FM ADPCM MIDI オートベンド
@Bn1,n2,dly,bnd

・n1:-8192～+8191　　開始ディチューン値(半音≒683)
・n2:-8192～+8191　　終端ディチューン値(半音≒683)
・dly:0～32767　　遅延時間
・bnd:-32767～+32767　　(正)変化所用時間,(負)継続時間,(0)いずれも設定しない

@Bの後ろに2つ以上のパラメータを書くとオートベンドの設定となる。

パラメータn1,n2の変化レートが異なるだけで機能自体は @Kn1,n2,dly,bndと同じ。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

FM ADPCM MIDI ポルタメント
(k1,k2)n,dly,prt
[k1,k2]n,dly,prt

・k1:c,d,e,f,g,a,b　　ポルタメント開始音階
・k2:c,d,e,f,g,a,b　　ポルタメント終了音階
・n:1～32767　　絶対音長
・dly:0～32767　　遅延時間
・prt:-32767～+32767　　(正)変化所用時間,(負)継続時間,(0)いずれも設定しない

ポルタメントを行う。

k1はポルタメント開始音階MML。

k2はポルタメント終了音階MML。

このほか、以下のMMLを(),[]内に記述できる。

調号	# + - !
音長	1～(付点も可)
絶対音長	*1～*32767
ｵｸﾀｰﾌﾞｺﾏﾝﾄﾞ	0-1～09
ｵｸﾀｰﾌﾞｽｲｯﾁ	< >

和音コマンドとは違い、オクターブコマンド、オクターブスイッチは(～),[～]外へも影響する。

オートベンドとポルタメント

オートベンドとポルタメントの具体例と両コマンドの対象デバイスにおける効果の相違点について少し解説します。

(c+4<<c+),8

はc+をディレイで指定した8カウント分演奏したあと、4分音符-8カウントの時間内に2オクターブ上のc+まで上昇します。オクターブスイッチの影響で以降2オクターブ上になることに注意してください。

(g,e)24&e

は絶対カウント24の時間内にgからeへピッチダウンを行いキーオフせずeへつなぐことになります。

ここで注意がひとつあります。以上2つの例はFM音源においてのみ有効です。1つ目が実現不可能なのは、MIDIでは楽器側のベンドレンジが1オクターブと想定して処理しているため1オクターブを超えたポルタメントは不可能だからです。

内蔵FM音源は音程がリニアな構造をしているため発音可能範囲内であれば任意の音程を出力可能ですが、MIDIにおいてはポルタメントは(オートベンドも)ベンダーホイールの上げ下げでピッチをコントロールしているため2つ目の例をMIDIに対して行うとFM音源とは違った動作を示してしまいます。2つ目の例をFM音源と同じ効果をMIDIで実現したい場合は、

(g,e)24&g

ということになります。つまりMIDIにおいてのタイはキーオフせず、ベンダー値を以前のまま保持することを意味しているのです。同様に、

o4 (c<c)&c

はFM音源ではo4のcから1オクターブ上のo5のcへポルタメントを実行しo5のcへ音をつなぐことができますがこれと同じことをMIDIで実現するためには、

o4 (c<c)&>c

あるいは、

o4 (c<c)&o4c

としなければなりません。MIDIの場合は発音後どれくらいベンダーホイールを動かしたかで音程をコントロールするのでタイの場合はポルタメントの開始音を書かなければならないのです。

このような相違点はオートベンドコマンドにも見られます。

@b0,683,0 c4　　　(683≒8192/12≒1半音)

ではFM音源ではc4がc+4へ向かって上昇します。そしてc+4を持続したい場合、FM音源の場合ならば、

@b0,683,0 c4&c+4

とすればよいのですがMIDIの場合は、

@b0,683,0 c4&c4

としなければならないわけです。

ところがMIDIパートも前述のタイモード設定コマンド@Jで@J1としてFM音源互換モードに設定すれば、MIDIパートでもまったくFM音源と同じ感覚でタイ/スラーを使用することができます。

音長は(～),[～]の外に絶対音長で書くこともできる。外に書いた場合は'*'をつけるつけないに関わらず絶対音長と見なされる。絶対音長指定ではnは0≦n≦32767まで設定可能なので全音符以上の発音時間を設定することもできる。

音長は省略するとデフォルトの音長(Lまたは@Lコマンドで指定したもの)が起用される。

dlyはディレイで絶対音長単位でポルタメントが始まるまでの遅延時間を設定することができる。設定可能範囲は0≦dly≦32767。

prtは正値の場合はポルタメントにかける時間を表す。負値の場合はポルタメントを完了してからどのくらいの時間、終了音高を継続させるかの設定となる。0にするとディレイ以後、発音する音符の音長のすべてを使って目的の音高までポルタメントを実行する。

dlyとprtはともに省略可能。省略時は以前の設定値が起用される。

MIDI楽器において、後述の@Gコマンドでベンドレンジを12(1オクターブ)以外に変更したとき、あるいはMIDI楽器のベンドレンジが12に設定されていないときの動作は表記通りにならない。

例1

```
o4(g2<g)
o4[g2<g]
```

オクターブ4のgで発音し、2分音符分鳴り終えるまでにオクターブ5のgまで滑らかに音程を変化させていく。コマンド終了後はデフォルトオクターブは5になっていることに注意。

例2

```
(o3d*96,e),48
[o3d*96,e],48
```

オクターブ3のdで発音し、このピッチで48カウント分演奏し、それ以降絶対音長96で消音するまでにオクターブ3のeまで音程をシフトしていく。

例3

```
(o3eo4d)48,24,-8
[o3eo4d]48,24,-8
```

オクターブ3のeで発音し、24カウント分、元の音高を持続し、16カウント(=48-24-8)の時間内でオクターブ4のdまでポルタメントする。そしてオクターブ4のdの音高を8カウント分保持して消音する。

ポルタメント音符もQコマンドの影響を受ける点にも注意すること。たとえば、

```
q4 (b-<b-)
```

とした場合は1オクターブ上のb-に達する前にキーオフしてしまう。

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

> ### 2つのポルタメントコマンド(～),[～]の相違点
>
> Z-MUSIC ver.3.0から新設された[～]によるポルタメントはMIDIパートで、その違いが表れます。FM音源やADPCMパートではまったく同機能として働きます。
>
> MIDIパートでは、ある音程から目的の音程へ変化させる場合、始まりの音程を基準にするか、目的の音程を基準にするかを選ぶことができます。
>
> たとえば、
>
> (c,e)
> [c,e]
>
> というようなドからミへのポルタメントを考えた場合を見てみます。
>
> 前者(c,e)ではcを発音後、これの音程を徐々に上げていき、目的のeの音程に達したときに処理を止めます。つまりeの音はcの音を3度上げた音なのです。これを分解して表記すると(半音≒683とする)、
>
> @b0c& .... @b2732c
>
> のようになります。
>
> これに対して後者[c,e]は、目的の音程eを3度下げたcの音程で発音し、音程を上げてe本来の音程に達すると処理を停止します。こちらは分解して表記すると(半音≒683とする)、
>
> @b-2732e& .... @b0e
>
> のようになります。
>
> これらになんの違いがあるかは楽器の性質に依存します。たとえばPCM系の音源の場合は、サンプリングしたベースノートの音高付近がもっともリアルに聴こえますのでポルタメント前後でもっとも長く鳴らす音程がベースノートに近くなるようにする、などの細かい工夫が行えます。

MIDI　ベンドレンジ変更

@Gn

・n:0-127

ベンドレンジの設定を行う。nの有効範囲はMIDI楽器によって異なる。また、外部からベンドレンジを操作できないものや、プログラムチェンジ(音色切り換え)時に初期化されてしまうものなどがあるので、手持ちの楽器のマニュアルをよく読み、仕様を理解してから使用すること。ベンドレンジを変更すればオートベンドコマンドを用いて1オクターブ以上のベンドを行うこともできるが、ポルタメントの効果も変化してくるので注意すること。

初期値はn=12(1オクターブ)を設定しているが、楽器側がベンドレンジを変更できないものであれば不定である。

FM音源パートやAD PCM音源パートでは常にn=12(1オクターブ)に固定。

■パンポット

FM ADPCM MIDI　パンポット

Pn

・n:0～3

パンポットの設定を行う。設定範囲は0≦n≦3。初期値は3。

0は出力停止、1は左、2は右、3は中央。

FM音源ではP0はミュートに相当するがMIDIではこれに対応する機能がないので便宜上@V0 @U0を行う。通常はMIDIではP0は用いないこと。

例

```
P1 cde P2 fga P3 b
```

(cdeが左から,fgaが右から,bが中央から演奏される)

FM ADPCM MIDI　パンポット

@Pn

・n:0～127

パンポットを多段階指定する。一応MIDI専用だがFM/AD PCMに設定した場合は値に応じて3段階の左/中央/右に変換される。

設定範囲は0≦n≦127であるがMIDI楽器によっては無効となる場合がある(M1,SY77など)ため、各MIDI楽器のマニュアルでパンポットが可変かどうか確認すること。

FM ADPCM MIDI　相対パンポット

@P±n

・n:-127～+127

パンポット(@P)を相対的に設定する。設定範囲は-127≦n≦+127で、演算の結果、オーバーフローを起こした場合は0または127へ修正される。

例

```
@p64|:10 @p+5 c16 :|
```

(音が中央からだんだん右へ鳴るようになっていく)

> ### パンポット
>
> ローランドMT-32系(CM-32L, CM-32P, CM-64L, CM-500)のパンポットは実はMIDI規格とは正反対の値で音場が切り換わります。よってP1,P2,P3はそれぞれMT-32系においてはR,M,Lと音場が切り換わります。
>
> Z-MUSICはMIDI規格に従って左を@P0、中央を@P64、右を@P127としています。
>
> FM音源やその他のMIDI楽器と同じ音場を得るためにMT-32系をご使用の方は背面のアウトプットを左右入れ換えて接続することをおすすめします。

■テンポ

FM ADPCM MIDI　テンポ

Tn

・n:1～32767

テンポを設定する。設定範囲は使用タイマやマスタークロックの値により変動する。

nは1分間の4分音符の数に相当する。

初期値は120。ひとつのトラックで指定すると全トラックに影響する。

FM ADPCM MIDI　相対テンポ

T±n

・n:-32768～32767

テンポを相対的に設定する。演算の結果有効範囲を超えてしまった場合は設定可能範囲内に自動修正される。
ひとつのトラックで指定すると全トラックに影響する。

FM ADPCM MIDI タイマ設定
@Tn

・n:0〜16383

割り込み周期を割り込みタイマへ直接設定する。機能的にはTコマンドと同じだがパラメータnが割り込み周期である点が違う。通常の音楽制作では使用しない。

以下に各タイマとテンポの相関を表す。(マスタークロック=192のとき)、

タイマA＝1024−(78125/テンポ)
タイマB＝256−(78125テンポ)/16
MIDIタイマ=30,000,000/(192*テンポ)

ひとつのトラックで指定すると全トラックに影響する。

FM ADPCM MIDI 相対タイマ設定
@T±n

・n:-32768〜32767

タイマ値を相対的に設定する。@T同様通常の音楽制作には用いない。演算の結果有効範囲を超えてしまった場合は設定可能範囲内に自動修正される。

ひとつのトラックで指定すると全トラックに影響する。

■音色/バンク切り換え

FM ADPCM MIDI 音色切り換え
@n 音色番号

・n:1〜32768

音色を音色番号nに設定する。初期値なし。

MIDIでは129以上の値を設定すると音色バンク切り換えのメッセージも同時に送出される。

バンク上位値=((n >> 14).and.$7f)
バンク下位値=((n >> 7).and.$7f)

FM ADPCM MIDI 音色切り換え
@n1,n2

・n1:0〜16383(255) バンク番号
・n2:1〜128 音色番号

音色を、バンクn1,音色番号n2に設定する。初期値なし。

FM音源、AD PCM音源パートのバンクは0-255が有効範囲。

MIDIでは音色バンク切り換えのメッセージも同時に送出される。

バンク上位値=((n1 >> 7).and.$7f)
バンク下位値=(n1.and.$7f)

FM ADPCM MIDI 音色切り換え
@n1:n2,n3

・n1:0〜127 バンク上位
・n2:0〜127 バンク下位
・n3:1〜128 音色番号

音色を、バンクn1:n2,音色番号n3に設定する。初期値なし。

FM音源、ADPCM音源パートのバンクはn1*128+n2で与えられる。

MIDIでは音色バンク切り換えのメッセージをn1を上位バンク値として、n2を下位バンク値として送出する。

FM ADPCM MIDI バンク切り換え
In1,n2

・n1:0〜127 バンク上位
・n2:0〜127 バンク下位

音色のバンク切り換えを行う。

KORGのWAVESTATION、ROLANDのGS音源などでは音色がバンクごとに分かれており、音色切り換えはこの命令でバンクを指定してから行う。同一バンク内で切り換えを行う場合は一度設定すれば以降設定する必要はない。

n1はバンク番号上位、n2はバンク番号下位を表す。パラメータは省略すると強制的に0になる。

GS音源ではバンク切り換えパラメータの上位のみ有効としているので

In1

のみでバンク(GS音源ではバリエーションと呼んでいるが)n1へ切り換えることができる。

WAVESTATIONではバンク切り換えパラメータの下位のみ有効としているので、

I0,n2

でバンクn2へ切り換えることができる。

例

```
i0 @128
i0,12 @11
```

FM音源,AD PCM音源パートのバンク

FM音源パートの音色はリニアに1〜32768まで存在しこれらを直接任意の音色に切り換えることができます。しかし、MIDIパートと同じように音色番号を1〜128までとし、これらが0〜255のバンクに分かれていると想定して取り扱うことも可能です。たとえば、

@221

はユーザーの好み次第で、

i0,1 @93

とすることもできます(0*16384+1*128+93=221)。

AD PCM音源パートの場合は、後述の[PCM_MODE]の設定によって扱いが多少異なります。[PCM_MODE TIMBRE]のときはFM音源/MIDIパートと同じで、0〜255のバンク、1〜128の音色番号で管理することができます。

一方[PCM_MODE TONE]ではバンクの設定はバンクという概念はなく、音色番号1〜256でトーンセットを切り換えます。MMLでは@1〜256はもちろん、I0〜255でも便宜上切り換えることができます。

■アフタータッチシーケンス

FM ADPCM MIDI アフタータッチシーケンス
@Zn1,…,n8

・ni:0〜127

アフタータッチを音長の1/8時間単位で設定する。これをZ-MUSICではアフタータッチシーケンスと呼ぶ。発音後の音のニュアンスを細かく変化させることができる。たとえば、

@z127,120,110,127,80,60,50,10

として絶対音長192の音長でキーオンしたとすると、キー音直後から24(=192/8)カウントごとに、アフタータッチが127,120,110,...と変化していく。

MIDI楽器によっては発音後の鍵盤を押す強さ(即ちアフタータッチ)によって音色のニュアンス等を変えることができ、サンプラーなどでは全く別の音色を発音させることも可能なので使用方法によってはかなり興味深い効果を得ることができる。

FM音源やADPCM音源パートでこの機能を使用した場合は発音後の絶対音量を設定値で変化させることができる。ソフトエンベロープなどを実現したい場合に威力を発揮する(もちろんMIDI楽器の設定でアフタータッチに対応する効果を音量に設定しておけばFM音源パートなどと同じようにソフトエンベロープ的な効果を得ることができる)。

パラメータは任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能である。たとえば、

```
@z127,,,10
@z,,,,120,,1
@z,,120,,-10,+5
```

といった記述が可能である。省略した部分は前の値を持続するということになり、その間はアフタータッチの情報は送信されない。

相対指定時にオーバーフロー/アンダーフローを起こした値は有効範囲内の最大値,最小値に修正される。

値をすべて省略した場合はアフタータッチ・シーケンス解除と解釈される。

例

```
@z
```

AD PCM音源部で使用可能となるのはMPCM.X組み込み時のみである。

■ビブラート(ピッチモジュレーション)

FM ADPCM MIDI ビブラートデプス
@Mn1,…,n8 ...(1)
または
@Mn ...(2)

・n,n1〜n8:-32768〜32767

ピッチモジュレーション（音程を細かく震わせる）の振幅設定を行う。

(1)1/8モード　ピッチモジュレーション

発音後、音長の1/8時間単位で振幅(モジュレーションデプス)を変化させることができる。設定可能範囲はFM音源は-32768≦n1～n8≦32767、MIDIのノーマルモードは0≦n1～n8≦127、拡張モード時は-768≦n1～n8≦768(-8192≦n1～n8≦8191)。

値は任意の位置で省略可能、8個まで記述でき、相対指定も可能で、

@m60,,,50,127
@m,,,,127
@m,,120,-10,+5

といった記述が行える。

1/8モードを指定した場合は後述のディレイコマンドで設定したディレイ値は無視される。

(2)ディレイモード　ピッチモジュレーション

振幅(モジュレーションデプス)をひとつだけ設定する。振幅の設定範囲は－32768≦n≦32767。

後述のディレイコマンドで設定したディレイのあとｎの深さでピッチモジュレーションを実行する。

初期値はいずれのモードにおいても0。またすべての値を省略した場合はモジュレーション解除のスイッチとなる。

例

@m

■ARCC(Assignable realtime control change)

FM ADPCM MIDI　ARCCコントロール設定

@Cn,r,m

・n:0～127　　コントロール番号
・r:0～127　　リセット値
・m:0～127　　波形基準値

任意のコントロールチェンジｎを選び、これを本命令で登録しておくと、後述の@Ａコマンドにて、そのコントロールチェンジで音源をリアルタイムにコントロールすることができる。これをZ-MUSICではアサイナブルリアルタイムコントロールチェンジ(Assignable realtime control change)と呼ぶ(以後ARCC)。

MIDIの場合はコントロールチェンジ番号の有効値はMIDI楽器によって異なるので各楽器のマニュアルのインプリメンテーションチャートを参考のこと。

FM音源では独自の、

$80=PANPOT	(0-127)	*1
$81=PMS	(0-7)	
$82=AMS	(0-3)	
$84=AMD	(0-127)	
$85=PMD	(0-127)	
$86=LFO SPEED	(0-127)	*2
$87=NOISE	(0-63)	

が対応している。なお()内の数値は各コントロールの有効範囲を表している。

*1　パラメータ的には0-127で行われるが実際の効果は左、中央、右の３段階にしかならない

*2　FM音源のレジスタには0-127の2倍の値(0-254)が書き込まれる。

AD PCMにおいてはコントロール番号はMIDIに準拠しているものの、現在では以下の、

7=VOLUME 10=PANPOT 11=EXPRESSION

の3つのみしかサポートされていない。

ｒは、ARCCの実行を解除したときに登録したコントロールチェンジｎに対して行うリセット値。省略時は０となる。

ｍは、後述の拡張ARCCモード使用時に使用可能となるプリセット波形やユーザー波形の基準値となる値(拡張MML[ARCCn.WAVEFORM]を参照)。

ADPCM MIDI　ARCCデプス

@An1,…,n8 ...(1)

または

@An ...(2)

・n,n1～n8:-128～127

(1)１/８モード　ARCC

発音後音長の1/8時間単位で先述の@Cで登録したコントロールチェンジへ値を送信し音源を制御することができる。値の設定範囲は0≦n1～n8≦127（拡張ARCCモード時には設定範囲は-127≦n1～n8≦127）。

値は任意の位置で省略可能、8個まで記述でき、相対指定も可能で、

@a60,,,50,127
@a,,,,127
@a,,120,-10,+5

といった記述ができる。

こちらのモードを指定した場合は後述のディレイ・コマンドで設定したディレイ値は無視される。

(2)ディレイモード　ARCC

ARCCの設定値をひとつのみ設定する。設定範囲は0≦n≦127（拡張ARCCモード時には設定範囲は-127≦n≦127）。

後述のディレイコマンドで設定したディレイのあと、ｎの値で@C命令で登録したコントロールチェンジを実行する。

初期値はいずれのケースにおいても0。また、すべての値を省略した場合はARCC解除のスイッチとなる。

例

@a

■アンプリチュードモジュレーション（トレモロ）

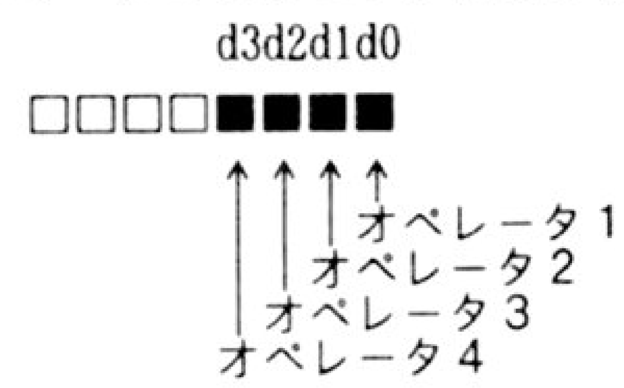
FM　トレモロ操作対象オペレータ設定

@Cm

・m:%0000～%1111

X680x0のFM音源は４つのオペレータ（発振器）によって構成されているがその1～4の４つのオペレータのうち、どのオペレータに対してアンプリチュードモジュレーションの処理を施すかを設定できる。特殊な効果(たとえばワウワウ効果)を表現したいときなど以外は特に設定する必要はない。

パラメータのｍはビット構成を取っており、



ビット＝1に対応するオペレータに対してモジュレーション処理を施す。

なお、このコマンドで設定した設定値は次回の音色切り換えまで有効である。音色切り換え時にはその音色のキャリアオペレータに対してモジュレーション処理を施すような設定値が自動的に設定される。

本命令はFM音源に対しての知識を必要とする。

例

@c%1001（オペレータ１と４にモジュレーション処理をする）

FM　トレモロデプス

@An1,…,n8 ...(1)

または

@An ...(2)

・n,n1～n8:-128～127

アンプリチュードモジュレーション（音量を細かく震わせる）の振幅設定を行う。

(1)１/８モード　アンプリチュードモジュレーション

発音後音長の1/8時間単位で振幅(モジュレーションデプス)を変化させることができる。値の設定範囲は-127≦n1～n8≦127。値は任意の位置で省略可能、8個まで記述でき、相対指定も可能で、

例

@a60,,,50,127
@a,,,,127
@a,,120,-10,+5

といった記述ができる。

こちらのモードを指定した場合は後述のディレイコマンドで設定したディレイ値は無視される。

(2)ディレイモード　アンプリチュードモジュレーション

振幅をひとつのみ設定する。設定範囲は-127≦n≦127。後述のディレイコマンドで設定したディレイのあと、nの振幅でモジュレーションを開始する。

初期値はいずれのケースにおいても0。また、すべての値を省略した場合はアンプリチュードモジュレーションの解除スイッチとなる。

例

@a

FM ADPCM MIDI ディレイ
@Hn1,n2
・n1:0～32767
・n2:0～32767
FM音源のピッチモジュレーション、アンプリチュードモジュレーション、またはMIDIのピッチモジュレーション、ARCC、拡張ピッチモジュレーション、拡張ARCCのディレイモード用のディレイを設定する。
n1はピッチモジュレーション用のディレイ。
n2はアンプリチュードモジュレーション/ARCC用のディレイ。
値はどちらか一方の省略は可能。初期値はともに0。

例

@h24,10 (両方同時に設定)
@h,10 (アンプリチュードモジュレーション/ARCC/拡張ARCCのディレイのみ設定する)
@h24 (ピッチモジュレーションのディレイのみ設定する)

FM ADPCM MIDI スピード
@Sn1,n2
・n1:1～32767
・n2:1～32767
FM音源のピッチモジュレーション、アンプリチュードモジュレーション(またはMIDIの拡張ピッチモジュレーション、拡張ARCCモード時)のモジュレーションスピードを設定する。設定範囲は1≦n1,n2≦32767。
本命令はMIDIパートのノーマルモジュレーションモードではまったく考慮されないもので、MIDIパートにおいては拡張モードのときのみ有効となる。
実際の波形にどう影響するかは拡張MML[VIBRATO.WAVEFORM],[ARCCn.WAVEFORM]の項で解説するが、値が小さいほど高速に、値が大きいほど低速に音が振動する。
値はどちらか一方の省略は可能。初期値はともに6。

例

@s24,10 (両方同時に設定)
@s,10 (アンプリチュードモジュレーション/拡張ARCCのスピードのみ設定する)
@s24 (ピッチモジュレーションのスピードのみ設定する)

FM ADPCM MIDI 波形タイプ選択
Sn1,n2
・n1:0～4,8～32767 (5～7は未定義、使用不可)
・n2:0～4,8～32767 (5～7は未定義、使用不可)
FM音源のピッチモジュレーション、アンプリチュードモジュレーション(またはMIDIの拡張ピッチモジュレーション、拡張ARCCモード時)のモジュレーション波形をセレクトする。
n1がピッチモジュレーション用の波形、n2がアンプリチュードモジュレーション用(または拡張ARCC)の波形に対応する。
波形は波形番号0～4がプリセット波形で、波形番号8～32767がユーザー定義波形となる。スピードや振幅パラメータの波形への影響については拡張MML[VIBRATO.WAVEFORM],[ARCCn.WAVEFORM]の項で図解する。

プリセット波形

0:鋸歯波 1:矩形波 2:三角波 3:ワンショット鋸歯波 4:ランダム波
値はどちらか一方の省略は可能。初期値は共に2の三角波。

例

s1,1 (両方同時に設定)
s,2 (アンプリチュードモジュレーション/拡張ARCCの波形のみ設定)
s3 (ピッチモジュレーションの波形のみ設定)

FM ADPCM MIDI ﾓｼﾞｭﾚｰｼｮﾝ･ﾎｰﾙﾄﾞ(同期設定)
Hn1,n2
・n1:0～1
・n2:0～1
FM音源のピッチモジュレーション、アンプリチュードモジュレーション(またはMIDIの拡張ピッチモジュレーション、拡張ARCCモード時)のモジュレーション波形を、キーオンと同期して初期化させるか(同期モード)、キーオンと同期して初期化せずに波形タイプの切り換えが起こるまで継続(ホールド)させるか(非同期モード)を設定する。
n1がピッチモジュレーション用、n2がアンプリチュードモジュレーション用(または拡張ARCC)の同期／非同期設定パラメータとなる。
値はどちらか一方の省略は可能。初期値は共に0の同期モードが設定されている。

例

h1,1 (両方同時に設定)

波形メモリの効能

Z-MUSICではユーザーが定義したユーザー波形(波形メモリ)を使ってピッチモジュレーション(以下PM)、アゴーギク、アンプリチュードモジュレーション(以下AM)、ARCC、エンハンスドベロシティシーケンスを実行することができます。Z-MUSIC ver.2.0では、波形メモリを使用した場合、振幅値は意味を持ちませんでしたがver.3.0以降では、振幅値が有効となります。ピッチモジュレーション、アゴーギクでの原波形振幅値は256に相当し、効能値は、

波形値×振幅÷256

で求められます。また、アンプリチュードモジュレーション、ARCC、エンハンスドベロシティシーケンスでの原波形振幅値は32に相当し、効能値は、

波形値×振幅÷32

で求められます。よって、ver.2.0で波形メモリを用いた演奏データをver.3.0で演奏させた場合は、やや演奏のニュアンスが異なってくる場合がありえます。万が一そう感じた場合はver3.0を、

A>ZMSC3 -2

のように'-2'オプションをつけて常駐させてから演奏させるとver.2.0と同等の演奏が得られます。
以下に音源パート別にその効能を示します。

●FM/AD PCM音源パートのピッチモジュレーションで使う場合

波形データは半音を64とした値で処理されます。各値がそのチャンネルのディチューンと加算されて、それがその瞬間のピッチとして決定されます。たとえばディチューンが10のときに波形メモリのデータが0,5,－5のとき、振幅が原波形指定(=256)だったとすると10＋0＝10, 10＋5＝15, 10－5＝5がそれぞれその瞬間のピッチになります。

例

(t1) @m256 @s6 s8 @h40 順位相で波形メモリ８番をFM･PMで実行
(t1) @m-256 @s6 s8 @h40 逆位相で波形メモリ８番をFM･PMで実行

●FM音源/ADPCM音源パートのアンプリチュードモジュレーション(ARCC)で使う場合

波形データはそのチャンネルのボリューム値に加算され、その和がその瞬間の音量として決定されます。音量の範囲は0～127ですがこの範囲を越えた場合はこの範囲内に修正されます。たとえば音量が@V125のとき波形メモリのデータが、0,5,－5のとき、振幅が原波形指定(=32)だったとすると125＋0＝125,125＋5＝127(130)125－5＝120がそれぞれその瞬間の音量になります。

例

(t1) @a32 @s,6 s,8 @h,40 順位相で波形メモリ8番をFM･ARCCで実行
(t1) @a-32 @s,6 s,8 @h,40 逆位相で波形メモリ8番をFM･ARCCで実行

●MIDIパートの拡張ピッチモジュレーションで使う場合

拡張モジュレーションモード時にはFM音源パートと同様な効果を得ることができます。拡張モジュレーションモードには振幅を半音＝64にするか半音≒683(ベンドレンジ＝12のとき)にするかの２つのモードがありますが波形データはこの影響を受けます。たとえば波形データが683で、振幅が原波形指定(=256)だったとすると半音＝64モード時には683/64≒10半音分 の変化量になりますが半音≒683モード時には1半音分の変化量となります。用途に合わせて使い分けてください。

例

(t1) m1 @m256 @s6 s8 @h40
順位相で波形メモリ8番をMIDI･拡張PMで実行
(t1) m1 @m-256 @s6 s8 @h40
逆位相で波形メモリ8番をMIDI･拡張PMで実行

●MIDIパートの拡張ARCCで使う場合

各波形データはARCCニュートラル値(@Cの第3パラメータ)と加算して、その和がコントロールチェンジへ出力されます。応用次第で疑似ランダムオートパンポットやFM音源のような自由波形による音量モジュレーションなども実現できます。

例

(t1) m,1 @a32 @s,6 s,8 @h,40
順位相で波形メモリ8番をMIDI･拡張ARCCで実行
(t1) m,1 @a-32 @s,6 s,8 @h,40
逆位相で波形メモリ8番をMIDI･拡張ARCCで実行

h,1　（アンプリチュードモジュレーション/拡張ARCCの波形を非同期モードに設定する）

h1　（ピッチモジュレーションの波形を非同期モードに設定する）

■モジュレーションモード

MIDI　モジュレーションモード

Mp,a

・p:0～2

・a:0～1

MIDIパートにはモジュレーションのモードが２つあり、ひとつは、ある指定されたタイミングごとにMIDI楽器へ制御メッセージを送信するモードで、これをノーマルモードと呼ぶ。

もうひとつはリアルタイムに絶えずMIDI楽器へ制御メッセージを送るモードでこちらを拡張モードと呼ぶ。

ノーマルモードは楽器側のハードウェア的な処理によって特殊効果を実現するのでZ-MUSIC側の負荷は少ない。一方、拡張モードは絶えず制御メッセージを送るため多少の負荷はあるものの、FM音源部のようにMML Ｓコマンドでセレクトされる波形に従って制御できるためノーマルモードよりも豊かな情緒表現が行える。

本命令はピッチモジュレーションとARCCを、ノーマルモードと拡張モードのどちらで実行するかを決定するもの。

・p:0　ノーマルモードの設定。@Mで設定された値を指定のタイミングでコントロールチェンジ１番のピッチ・モジュレーションに書き込む。

　@Mで指定できる振幅の範囲は0≦@M≦127。

・p:1　拡張モード１の設定。設定された振幅の効用をFM音源の@Mと同じにして(@Mで指定される振幅は半音＝64ということになる)、Sでセレクトした波形でピッチモジュレーションを実現する。

　@Mで指定できる振幅の範囲は－768≦@M≦768。

・p:2　拡張モード２の設定。設定された振幅の値はそのまま14ビットのMIDIのピッチベンダー直値に該当する。この振幅で、Sコマンドでセレクトした波形でピッチモジュレーションを実現する（ベンドレンジ＝12のとき、半音≒683）。@Mで指定できる振幅の範囲は-8192≦@M≦8191。

・a:0　ノーマルモードの設定。@Aで設定した値を指定されたタイミングで書き込む。

・a:1　拡張モードの設定。@Aで－127～127までの振幅を設定すると、Sコマンドでセレクトした波形に従って生成される値を@Cで登録したコントロールチェンジに対して出力できる。

パラメータp,aはどちらか一方の省略は可能。初期値はp,aともに0（ノーマルモード）。

例

m2,1　（両方同時に設定）

m,1　（ARCCのモードのみ設定する）

m1　（ピッチ・モジュレーションのモードのみ設定する）

※注意

本命令を実行するとそれまで設定されていた振幅(@Mおよび@Aの値)は無効となる。

■特殊機能スイッチング

FM MIDI 特殊機能スイッチング

＝n

・n:0～%11111111111　　(0～2047)

ピッチモジュレーション、ARCC、オートベンド、アフタータッチ・シーケンス、ベロシティシーケンスの各機能のスイッチングを行う。

ｎは各ビットごとに意味を持ち、対応は以下の通りである。

```
10 9 8  7 6 5 4 3 2 1 0
■ ■ ■  ■ ■ ■ ■ ■ ■ ■ ■
                      └ピッチモジュレーション
                    ARCC1
                  オートベンド
                アフタータッチシーケンス
              ベロシティシーケンス
            エンハンスドベロシティシーケンス
          アゴーギク
        オートポルタメント
    ARCC2
  ARCC3
ARCC4
```

各ビットを０にするとオフ、１にするとオンとなる。

初期値は0（全機能ＯＦＦ）である。

例

=%00011　（ピッチモジュレーション／ARCC0をＯＮ。それ以外をOFF）

※注意　ARCC2～4はMML @Aや@Cでは制御できない。ARCC1～3の制御については後述。

スイッチングオフについて

各特殊機能の制御MMLにパラメータなしや0を設定するとその機能は停止することになります。たとえば @M または @M0 でピッチモジュレーション機能を解除することができます。またはスイッチングコマンド'='を使って制御することもできます。この２つのスイッチングには微妙な違いがあるので注意してください。

まず、前者の場合では、機能停止時になんらかの初期化措置を音源/楽器側に対して実行します。たとえばMIDIパートで@M0とした場合はモジュレーション処理を終了するともに楽器側へモジュレーションデプス0を送信しています。

一方後者の場合は純粋にZ-MUSICがモジュレーション処理を中断するだけで楽器に対しては、なんら特殊な処置をしません。ですから、たとえばMIDIで、

@h40 @m127 c1 @m d1

@h40 @m127 c1 =0 d1

とした場合を考えてみます。両者ともcがディレイ40クロック後にモジュレーションを開始します。そして、次に発音されるdが前者と後者では食い違いが出てきます。

前者ではｄはモジュレーションなしで演奏されます。しかし、後者のほうはZ-MUSICが「ディレイ40クロックのあとモジュレーションを開始」という処理を止めただけで前の音符のcのモジュレーションがそのまま残ることになりdはモジュレーションがかかったまま演奏されてしまうのです。

こうしたことから'='によるスイッチングは前の状態を保存しつつ、特殊機能を停止したいというときに用いるとよいでしょう。

特殊効果機能の併用について

Z-MUSICではさまざまな特殊機能を同時に複数使うことができますが、オートベンドとポルタメントは同種機能のため同時使用はできません。万一、オートベンド時にポルタメントを実行した場合にはポルタメントのみ機能し、オートベンドは実行されません。

しかし、これ以外の組み合わせは可能です。たとえばポルタメントする音に対してピッチモジュレーション、ARCC/アンプリチュードモジュレーションをかけてさらにアフタータッチシーケンスを実行する....といった複数の効果をひとつの音に対して同時にかけることも可能です。

■発音制御

FM ADPCM MIDI　ホールド(ダンパーペダル)

＠Dn

・n:0～1

ダンパーペダルの状態の設定を行う。設定範囲は0がダンパーペダルオフ、1がダンパーペダルオンである。ダンパーオンの状態では発音はされるがダンパーオフになるまでキーオフされない。

各楽器の最大同時発音数を超えたときには各楽器に処理が委ねられる。通常はいちばん過去に発音された音がキーオフされそのボイスを用いて新たな音がキーオンされることになる。

FM ADPCM MIDI

＠Rn　ノートオフ省略モード

・n:0～1

n=1とするとそのトラックは「ノートオフ省略モード」になり、以後ノートオフ処理をしない。リズムマシンや一部の楽器のリズムキットは発音すればキーオフを送らずともよいものがあり(そういった楽器には「ノートオフモード」とか「ノーサスティン」といったパラメータがあるはず)、こういった楽器やチャンネルに対して「ノートオフ省略モード」にするとZ-MUSICのノートオフ処理を省略でき、割り込み演奏処理がいくぶん軽くなる。

ノートオフ省略モードでは音符にゲートタイムという概念がなくなるため、MML Qの影響を受けずエンベロープを最後まで実行するようになる。そのため音を細かく区切ったりなどというニュアンスは再現不可能になる。

n=0でノートオフ処理をする、n=1でノートオフ処理をしない。n省略時は@R0の設定がなされたとみなされる。また初期状態も@R0が設定されている。

FM ADPCM MIDI　強制消音
`

強制的にキーオフを行う。そのトラック内において発音中の音をすべて消音(ミュート)する。ただし、ダンパーONやキーオフなしモードに設定されている場合はノートオフできない。

> **FM音源トラックの最大同時発声数**
>
> 内蔵FM音源の和音とダンパーにおいての同時発声数は最大8音です。ただし、その割り当ては動的ではなく静的です。たとえばチャンネル1から4和音発音した場合、チャンネル1,2,3,4が使用されます。もし、ほかのトラックでこれらのチャンネルを使用していた場合、競合が起こり正しく演奏されません。FM音源で1音以上の発音を行わせる場合には、この「チャンネルの競合」に注意してください。

■フェードイン/フェードアウト

FM ADPCM MIDI　フェードイン/フェードアウト
¥n

・n:-128〜127

フェードインまたは、フェードアウトを行う。

nは音量増減スピードを表し絶対値が大きいほど増減スピードが速くなる。

-128≦n≦-1　フェードイン
n=0　フェードイン/フェードアウトモード解除
1≦n≦127　フェードアウト

各トラックごとに設定可能なので、各トラック違った速度でフェードインやフェードアウトが実行可能。

各音源の特性の相違から均等な音量バランスではフェードアウトされない場合もある。

n=0のフェードイン/フェードアウト解除は処理そのものを中止するだけなので元の演奏状態に戻したい場合は音量などの再設定を行う必要がある。

■繰り返し/反復記号

FM ADPCM MIDI　繰り返し
|:n … |r1 … :| |r2 … :| :|　...(1)

または

|:n … | … :|　...(2)

・n,r1,r2:1〜65535

(1)のケース

|:n〜:|で囲まれたMMLをn回演奏する(n省略時はn=2が自動設定される)。

|ri〜:|を|:n〜:|の中に挿入すると、|riから:|までの演奏データを、繰り返しri回目にだけ演奏させることができる。|:〜:|はメモリのある限り何重まででも括ることが可能で複雑なシーケンスも可能。

> **多重ループ構造について**
>
> Z-MUSICでは多重の繰り返し構造が設定可能ですが、必ずネスト構造になっていなければ希望通りの繰り返しを得ることができません。
>
> たとえば、
>
> |: a |: b |1 c :||2 d e :|
>
> という記述では正常に演奏することができません。
>
> これは、
>
> |: a |: b |1 c :||2 d e :| :|
>
> としなくてはなりません。
>
> 多重ループを使用する際は|:に対応した:|が必要ということなのです。

各値の設定可能範囲は1≦n, ri≦65535である。

例

|:3 a |:2 b :| c:|　(abbc abbc abbcのように演奏される)
|:2 a |1 b :| |2 c:|　(ab ac のように演奏される)

(2)のケース

|riのriを省略した|を|:n〜:|の中に挿入すると|以降に書かれた演奏データは最後の繰り返し時には演奏はしないといった特殊例になる。

例

|:5 a | b :|　(ab ab ab ab aのように演奏される)

FM ADPCM MIDI　反復記号
[〜]コマンド

[DO] 〜 [LOOP]　囲まれた間を半永久的にループ演奏する。
[D.C.]　始めに飛ぶ(1回のみ有効)。
[D.S.]　[SEGNO]または[$]へ飛ぶ(1回のみ有効)。
[SEGNO](または[$])
[TOCODA](または[*])　[CODA]へ飛ぶ(1回以上有効)。
[CODA]
[FINE](または[^])　[D.C.],[D.S.]を実行したあとにここで演奏終了。

このほかに演奏データ作成を支援するコマンドがある。

これらのコマンドは共通コマンド(D1)のときのみ有効となる。(D0)のときは無視される。(MEASURE4 (Dn)コマンド参照)

[!]　次の[!]までジャンプする。ジャンプ後の音源やZ-MUSICの内部パラメータはそこまで通常に演奏を行ったときと同じになっている。
[!]が遇数個設定されなかった場合は、正常に動作しない。

[@]　次の[@]までジャンプする。ジャンプ後の音源やZ-MUSICの内部パラメータはそこまで通常に演奏を行ったときと同じになっている。
ほぼ[!]と同機能だが、あるひとつのトラックで設定すると他のトラックも同時にジャンプを開始する点が違う。[@]が遇数個設定されなかった場合は、正常に動作しない。

[end]　演奏を中断する。

> **[!]/[@]の動作**
>
> [!]/[@]の動作手順を以下に示します。
>
> 1. 音階／和音／ポルタメント以外の「発音しない」コマンドを実行
> 2. [!]/[@]を見つけるまで1. を繰り返す
> 3. [!]/[@]を発見
> 4. 通常演奏へ

■MIDI楽器制御命令

MIDI　MIDI楽器タイプ登録命令
@I n1,n2,n3

・n1:0〜127
・n2:0〜127
・n3:0〜127

Z-MUSICにそのトラックを演奏する楽器のメーカーID、デバイスID、モデルIDを設定する(所有しているMIDI楽器の各値にいては楽器のマニュアルを参照)。これは特に設定しなくても通常のコマンドの動作にはまったく影響しない。ただし、MMLの、

X
@E

に関しては楽器個別の動作をするために@Iで設定した情報を用いるため、これらのコマンドを使用する前には必ず設定する必要がある。

n1=メーカーID(例:ROLAND=$41, KORG=$42, YAMAHA=$43)
n2=デバイスID(各ユーザーが楽器に設定した値。特に設定した覚えのない場合はその楽器のデフォルトのデバイスIDを設定する)
n3=モデルID(例:MT-32=$16, SC55=$42 or $45)
n1,n2,n3すべて初期値なし。

MIDI　MIDIデータ送信
@X n1,…,xi

・xi:0〜255

MIDIデータの送信を行う。メーカーや楽器の種類には無関係でチェックサムなども発信しない。データはいくつでも記述可能。各値の設定範囲は

0≦xi≦255であるが、256以上の値を設定した場合は7ビットずつに分割されて送信される(8ビットでない点に注意。一般的なMIDI規格の慣例に従い、下位、上位の順に送信される。例:8192→$00,40)

本命令は楽器のメモリを書き換える場合もあるので、各自の責任の下で使用すること。

MIDI　ローランドエクスクルーシブ送信

Xn1,…,ni

・ni:0〜127

ローランド系の楽器にエクスクルーシブメッセージ送信を行う。@Iコマンドでメーカー IDにローランドを設定しておくことが前提。

データの転送方式は「ワンウェイコミュニケーションのデータセット」(コマンドID=＄12)。

データはいくつでも記述可能。各値の設定範囲は0≦ni≦127であるが、128以上の値を設定した場合は7ビットずつに分割されて送信される（一般的なMIDI規格の慣例に従い、下位、上位の順に送信される。例:8192→$00,40)。

ローランドエクスクルーシブに不可欠なチェックサム値は自動的に送信される。

本命令は楽器のメモリを直接操作するものなので、各自の責任の下で使用すること。

MIDI　汎用エフェクト設定

@Er,c,d

・r:0-127　(GS音源系ではリバーブ)

・c:0-127　(GS音源系ではコーラス)

・d:0-127　(GS音源系ではディレイ)

MIDI楽器のエフェクトを制御する。コントロールチェンジのエフェクト1(GSではREVERB)へrを、エフェクト3（GSではCHORUS)へcを、エフェクト4(GSではDELAY)へdを出力する。

ただし、楽器側がこれらのエフェクト制御に対応していない場合は、無意味。

パラメータは任意のものを省略できる。パラメータの有効範囲は0≦r,c,d≦127。初期値は不定。

MIDI　MT-32エフェクト設定

@En1,n2

・n1:1〜8

・n2:0〜1

@Iコマンドにて IDをローランドMT-32に設定してあることが前提。パートn1のエフェクトのスイッチを設定する。n1はパートナンバーを表し設定範囲は1≦n1≦8。n2はスイッチを表しn2＝0がOFF、1がONに相当する。

MIDI　NRPN設定

@Ya1,a2,d1,d2

・a1:0〜127

・a2:0〜127

・d1:0〜127

・d2:0〜127

MIDI楽器のＮＲＰＮを設定する。

MIDI楽器には、MIDI規格には定められていない機能レジスタを持つことが許されており、このコマンドはそのレジスタへの設定を行う。

a1,a2はそれぞれNRPNのレジスタ番号(アドレス)上位下位を表し、d1,d2はそれぞれデータの上位下位を表す。設定範囲はすべて0〜127でd1,d2のみ省略可能。

SC-55では音色パラメータの設定をここで行うことができる(SC-55マニュアルp75参照,SC-88VLマニュアルp72参照)。

■AD PCM制御命令

ADPCM　再生周波数設定

@Fn

・n:0〜6

内蔵AD PCM音源の再生周波数を変更する。

設定範囲は0≦n≦6で、各値は、

n=　0: 3.9kHz
　　1: 5.2kHz
　　2: 7.8kHz
　　3:10.4kHz
　　4:15.6kHz
　　5:20.8kHz　*1
　　6:31.2kHz　*1

に対応する。初期値は４で値省略時は４に自動設定される。

*1の周波数はAD PCMチップ供給クロックに対して改造を行ったX680x0に対してのみ有効。

■FM音源レジスタ設定

FM ADPCM　FM音源レジスタ設定

Ya,d

・a:0-255

・d:0-255

FM音源のレジスタａへ値ｄの書き込みを行う。

AD PCM音源部で使用した場合もFM音源のレジスタへの書き込みが行われる。使用にはFM音源に関しての知識と十分な理解が必要。

■ノイズ発生

FM　FM音源ノイズの設定

@On

・n:0-31

FM音源チャンネル８のオペレータ４(スロット32)からノイズを発生する。

nとノイズの周波数とは、

n=4000/(32×ノイズ周波数[kHz])

の関係があり、設定範囲は0≦n≦31。値を省略した場合はノイズモード解除と見なす。

■コントロールチェンジ

MIDI　コントロールチェンジ設定

Ya,d

・a:0-127

・d:0-127

コントロールチェンジa番に値dを書き込む。

パラメータa,dの設定範囲はともに0≦a,d≦127。コントロールチェンジについての詳細は各MIDI楽器のマニュアルのインプリメンテーション・チャートを参照。

■演奏制御

FM ADPCM MIDI　同期待ち

W

ほかのトラックから同期信号がくるまで演奏処理を一時中断する。

なお、本命令は設計の都合上、[@],[!],[JUMP]といったジャンプ系コマンドと併用できない点に注意。

FM ADPCM MIDI　同期信号送信

Wn

・n:1〜65534

同期待ちしているトラックに対して同期信号を送る。設定範囲は1≦n≦65534。同期信号を送られたトラックは直ちに演奏を再開する。

なお、本命令は設計の都合上、[@],[!],[JUMP]といったジャンプ系コマンドと併用できない点に注意。

例

```
(t1)   @1 v15 o4 q8 cdef w2  ←トラック２へ同期送信
(t2) w @1 v15 o4 q8 gab<c
       ↑どこからか同期信号がくるまで演奏を中断。
```

この例では結果的にはcdefgab<cが演奏される。

FM ADPCM MIDI　演奏強制再開

Jn

・n:1〜65535

トラックｎを強制的に初めから演奏開始にする。演奏が終了していても、

演奏中であっても演奏開始を受けたトラックは最初から演奏を始める。設定範囲は1≦n≦65535。

例

```
(t1)  @1 v15 o4 q8 j2r2j2
(t2)  @1 v15 o4 q8 l8cdef
```

この例では結果的にはcdef cdefが演奏される。

■コメント

FM ADPCM MIDI　コメント

/str

・str:文字列

コメントを与える。/以降その行の改行までをコメントと見なしコンパイル時には無視される。またコメントはコンパイル後のZMDには埋め込まれない。

例

```
/This is a pen.
```

■特権命令

FM ADPCM MIDI　ZMD埋め込み

;n

・n:0-255

直接ZMDコードｎを埋め込む。使用にはZMDの十分な理解を必要とする。使用方法によっては最悪、暴走してしまうので注意。設定範囲は0≦n≦255。

FM ADPCM MIDI　ワーク直接書き換え

?a,d

・a:-2147483648～2147483647

・d:0～4294967296 または -2147483648～2147483647。

そのトラックの演奏ワークのa番に値dを書き込む。dは数値の頭に±をつけて相対値の設定も可能。

使用にはZ-MUSICのワークエリアについての十分な理解を必要とする。使用方法によっては最悪、暴走してしまうので注意。

5．3．　拡張MML

ここではZ-MUSIC ver.3.0になって新たに追加されたMML群について解説する。ver.3.0になって新たに追加されたMMLは、

[～]

で囲まれた中に英単語文字列を入れる形態を取っている。テキスト音楽ソースの可読性を高めることが狙いでこうなった。なお、ほとんどの基本MMLコマンドがこの形態にも存在する。音楽ソース(ZMS)の可読性向上を図るならば、なるべくこの形式を利用するようにしたほうがよい。

なお、ver.2.0以前から存在した[～]形態のコマンド([DO],[LOOP]など)については5．2．節で解説してあるのでそちらを参照のこと。

■音色

FM ADPCM MIDI　音色切り換え

[TIMBRE n]
[PROGRAM n]

・n:1～32768

音色を音色番号ｎに設定する。初期値なし。基本MML @とほぼ同等の機能(相違点については囲み参照)。

MIDIでは129以上の値を設定すると音色バンク切り換えのメッセージも同時に送出される。

　バンク上位値=((n >> 14).and.$7f)
　バンク下位値=((n >> 7).and.$7f)

例

　[TIMBRE 25]（音色番号25を設定）

FM ADPCM MIDI　音色切り換え

[TIMBRE n1,n2]
[PROGRAM n1,n2]

・n1:0～16383　(FM音源AD PCM音源パートでは0～255)

・n2:1～128

音色を、バンクn1,音色番号n2に設定する。初期値なし。基本MML @とほぼ同等の機能(相違点については囲み参照)。

FM音源、AD PCM音源パートのバンクは0-255が有効範囲。

MIDIでは音色バンク切り換えのメッセージも同時に送出される。

　バンク上位値=((n1 >> 7).and.$7f)
　バンク下位値=(n1.and.$7f)

例

　[TIMBRE 4,25]　　（バンク4の音色番号25を設定）

FM ADPCM MIDI　音色切り換え

[TIMBRE n1:n2,n3]
[PROGRAM n1:n2,n3]

・n1:0～127　　バンク上位

・n2:0～127　　バンク下位

・n3:1～128　　音色番号

音色を、バンクn1:n2,音色番号n3に設定する。初期値なし。基本MML @とほぼ同等の機能(相違点については囲み参照)。

FM音源、AD PCM音源パートのバンクはn1*128+n2で与えられる。

基本MML @と拡張MML [TIMBRE]の相違点について

MIDIパートおよびAD PCMパートにおいては、両コマンド間に相違点はまったくありませんが、FM音源パートにおいては微妙な違いがあります。基本MML @による音色切り換えでは音色パラメータのハードウェアLFO、パンポットまでも音色パラメータとして更新してしまいますが、[TIMBRE]ではこれら2のパラメータは以前の状態を保持し、音色の本質的なパラメータのみを切り換えます。たとえば、

```
/          AR DR SR RR SL  OL KS ML D1 D2 AM      SQU
 (@1       31  0  0 15  0  25  0  2  0  0  0
           31 29  0 15  0  35  0  3  0  0  0
           31 29  0 15  0  35  0  5  0  0  0
           31 29  0 15  0   0  0  1  0  0  0
/          AL FB OM PAN
            5  7,15,  3)
.Assign tr1,FM1
(t1)p1 @1 c
(p)
```

とした場合はｃは、p1(左)ではなく@1により音色パラメータのPAN=3が設定された関係でp3(中央)で演奏されます。これに対して、

```
/          AR DR SR RR SL  OL KS ML D1 D2 AM      SQU
(@1        31  0  0 15  0  25  0  2  0  0  0
           31 29  0 15  0  35  0  3  0  0  0
           31 29  0 15  0  35  0  5  0  0  0
           31 29  0 15  0   0  0  1  0  0  0
/          AL FB OM PAN
            5  7,15,  3)
.Assign tr1,FM1
(t1)p1 [timbre 1] c
(p)
```

とした場合はｃは、[TIMBRE 1]が実行されてもパンポットのパラメータは更新されないので、これ以前の設定p1(左)で演奏されます。

MIDIでは音色バンク切り換えのメッセージをn1を上位バンク値として、n2を下位バンク値として送出する。

例

[TIMBRE 4:4,25]　　(バンク4:4の音色番号25を設定)

FM ADPCM MIDI　バンク切り換え
[TIMBRE_BANK n1,n2]
[PROGRAM_BANK n1,n2]

・n1:0〜127　　バンク上位
・n2:0〜127　　バンク下位

音色のバンク切り換えを行う。基本MML Iと同等の機能。

n1はバンク番号上位、n2はバンク番号下位を表す。パラメータは省略すると強制的に0になる。

GS音源ではバンク切り換えパラメータの上位のみ有効としているので、

[TIMBRE_BANK n1]

のみでバンク(GS音源ではバリエーションと呼んでいるが)n1へ切り換えることができる。

WAVESTATIONはバンク切り換えパラメータの下位のみ有効としているので、

[TIMBRE_BANK 0,n2]

でバンクn2へ切り換えることができる。

例

[TIMBRE_BANK 5] @128　　(バンク5,0の音色番号128に設定)
[TIMBRE_BANK 0,12] @11　(バンク0,12の音色番号12に設定)

FM ADPCM MIDI　音色振り分け
[TIMBRE_SPLIT sw b1,t1,m1,n1, b2,t2,m2,n2, ... b8,t8,m8,n8]
[PROGRAM_SPLIT sw b1,t1,m1,n1, b2,t2,m2,n2, ... b8,t8,m8,n8]

・sw:OFF,ON
・b1〜b8:0-16383(あるいは00:00〜127:127)
・t1〜t8:1-32768
・m1〜m8:0-127,128　　または　0-1C〜09G,C-1〜G9
・n1〜n8:0-127,128　　または　0-1C〜09G,C-1〜G9

発音する音高に応じて、音色を自動的に切り換える機能TIMBRE SPLITの設定を行う。ノート番号0〜127(0-1C〜09G)までを最大8つのブロックに分割し、ひとつのブロックごとに別々の音色を対応づけることができる。実際の演奏時には、発音する音符のノート番号が特定のブロックに属する場合は、その対応づけられた音色に切り換えてから発音を行う。

・sw:スイッチ

TIMBRE SPLIT機能の制御を行う。sw=OFFで機能解除、sw=ONで機能有効に設定する。

省略して、これ以降のパラメータ(b1〜b8,t1〜t8,m1〜m8,n1〜n8)を記述した場合はONが指定されたと見なす。またOFFを指定した場合には以降のパラメータを記述する必要はない。なお、swに数値は与えられない点に注意。

・b1〜b8:バンク番号

切り換える音色のバンク番号を設定する。設定範囲は0≦b1〜b8≦16383。

また、mm:nnのような表記で指定することも可能でこのときはmmがBANK MSB,nnがBANK LSBに対応し、設定範囲は0≦mm≦127,0≦nn≦127。

・t1〜t8:音色番号

切り換える音色番号を設定する。設定範囲はFM音源,AD PCMパートでは1≦t1〜t8≦32768。

MIDIパートでは1≦t1〜t8≦128。

・m1〜m8:範囲開始ノート番号

分割ブロックの開始ノート番号を設定する。設定範囲は0≦m1〜m8≦127。

MMLを使って0-1C〜09G、C-1〜G9といった指定も可能。

・n1〜n8:範囲終端ノート番号

分割ブロックの終端ノート番号を設定する。設定範囲は0≦m1〜m8≦127。

MMLを使って0-1C〜09G、C-1〜G9といった指定も可能。

例

[TIMBRE_SPLIT ON,0,32,o4c,o5c, 8,32,o2c,o3b]
　04C〜05Cの音符はバンク0,音色番号32で
　02C〜03Bの音符はバンク8,音色番号32で

演奏する。その他の音域の音符に関しては、それまでに設定されている音色で演奏する。

FM ADPCM MIDI　音色振り分けスイッチ
[TIMBRE_SPLIT.SWITCH sw]
[PROGRAM_SPLIT.SWITCH sw]

・sw:0,1　または　ON,OFF

TIMBRE SPLIT機能の制御を行う。sw=0で機能解除、sw=1で機能有効。

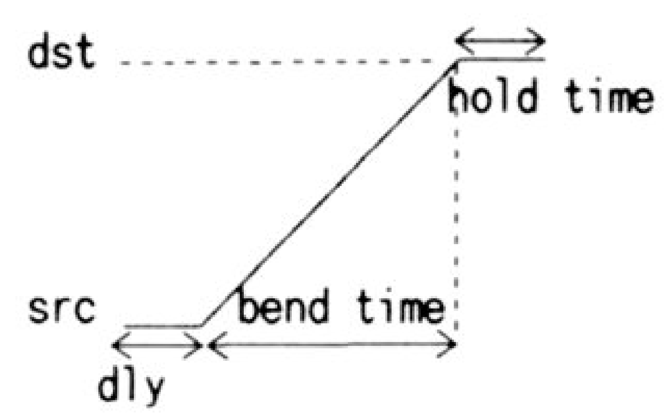
[TIMBRE_SPLIT]のswと同様に文字パラメータを与えることもできる。この場合はsw=OFFで機能解除、sw=ONで機能有効となる。

例

[TIMBRE_SPLIT.SWITCH off]　(TIMBRE_SPLITを機能解除する)

■キートランスポーズ/ディチューン/オートベンド

FM ADPCM MIDI　オートベンド
[BEND src,dst,dly,hold time(or bend time)]



・src:-16384〜+16383
・dst:-16384〜+16383
・dly:0〜32767
・hold time:-32768〜0　または　bend time:1〜32767

オートベンドの設定を行う。この設定以降に発音する音はこの設定の音程変化に従う。

まず、絶対音長dlyの間、ピッチsrcを持続し、目的のピッチdstに向けて音程を変化させていく。変化時間はhold time/bend timeにて行う。値を負にした場合はhold timeの設定となり、その絶対値の絶対音長分目的のピッチdstを持続するような音程変化を行う。逆に、正にした場合はbend timeの設定となり、その値の絶対音長の間に音程変化を行う。

機能的には基本MMLである@Bと同等のもので、1オクターブの変化量を8192としている。ただし、MIDIパートでのパラメータ有効範囲は -8192≦src, dst≦8191。

例

[BEND -65,765,24,48]

(ピッチ-65から765までの音程変化を行う。ピッチ-65を絶対音長24持続し、ピッチ765までの音程変化を絶対音長48内で行う)

[BEND 0,683,24,-48]　c*192

(ピッチ0から683までの音程変化を行う。ピッチ0を絶対音長24持続してから、ピッチ683までの音程変化を行い始める。そしてピッチ683の音程を絶対音長48持続する。この例では音程変化にかける時間は絶対音長120(=192-24-48)になる)

設定を解除するには、

[BEND OFF]

とするか後述の[BEND.SWITCH]コマンドを利用する。解除した場合も変化開始ピッチsrcはそのまま設定されているのでオートベンドを解除しつつピッチも変更したい場合は、

[BEND 0]

のように変化開始ピッチのみを設定し直す。

FM ADPCM MIDI　オートベンドのスイッチ
[BEND.SWITCH sw]

・sw:0,1　または　OFF,ON

オートベンドの動作のON,OFFを行う。swは0がOFF,1がONに対応する。また、文字列 ON,OFFを直接与えることもできる。

例

[BEND.SWITCH 0]　　[BEND.SWITCH OFF]　　(スイッチオフ)
[BEND.SWITCH 1]　　[BEND.SWITCH ON](スイッチオン)

MIDI　ベンドレンジ設定
[BEND.RANGE n]

・n:0-127

MIDI楽器に対してベンドの変化量領域を設定する。MIDIでは14ビット(-8192〜8191)のピッチベンドの変化を与えることができるが、この変化 -8192〜0, 0〜8191 が実際にどのような音程変化をもたらすかを設定できる。基本コマンドの@Gに同等のもの。初期値は12(1オクターブ)

例

[BEND.RANGE 12]

FM ADPCM MIDI　ピッチ指定(ディチューン)
[PITCH n]

・n:-8192〜+8191

ピッチベンダーホイールの値を設定する。機能的には基本MMLである@Bと同等のもので、1オクターブの変化量を8192としている。ただし、MIDIパートでのパラメータ有効範囲は-8192≦src,dst≦8191。

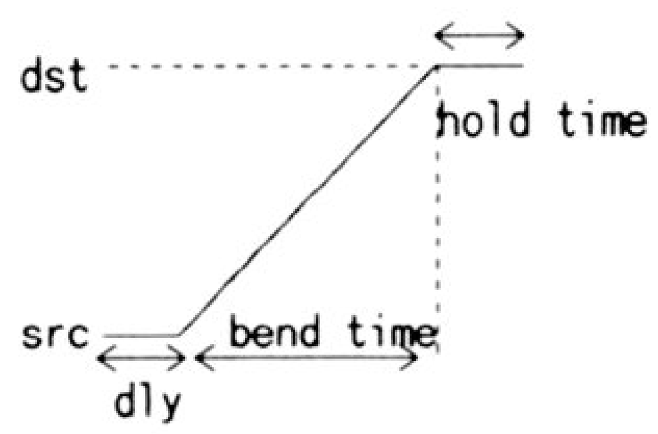
FM ADPCM MIDI　相対ピッチ指定(ディチューン)
[@PITCH n]

・n:-8192～+8191

ピッチベンダーホイルの値設定を相対指定で行う。機能的には基本MMLである@Bと同等のもので、1オクターブの変化量を8192としている。ただし、MIDIパートでのパラメータ有効範囲は -8192≦src,dst≦8191。

■ポルタメント

FM ADPCM MIDI　ポルタメント
[PUSH_PORTAMENT src,dest,step,gate,velo,dly,hold time(or port time)]
[PULL_PORTAMENT src,dest,step,gate,velo,dly,hold time(or port time)]
[PORTAMENT src,dest,step,gate,velo,dly,hold time (or port time)]



・src:o-1c～o9g
・dst:o-1c～o9g
・step:0-32767
・gate:0-32767
・velo:0-127　(相対指定-63～+63も可能)
・dly:0-32767
・hold time:-32768～0　または　port.time:1～32767

ポルタメントを行う。基本MMLの(～)によるポルタメントとほぼ同等。

・src:開始キー

ポルタメント開始ノート。オクターブ-1のcからオクターブ9のgまでのノートを指定することができる。

・dest:終端キー

ポルタメント終了ノート。オクターブ－1のcからオクターブ9のgまでのノートを指定することができる。

・step:ステップタイム

ポルタメント音符の音長を指定する。値は絶対音長指定で行い、その有効範囲は0～32767。

・gate:ゲートタイム

ポルタメント音符のゲートタイムを指定する。値は絶対音長指定で行い、その有効範囲は0～32767。

・velo:ベロシティ

ポルタメント音符の発音ベロシティを設定する。値の有効範囲は0～127で、数値前に±をつけて指定すれば相対指定となる。ただし、このときの有効範囲は-63～+63となる。

・ dly:ディレイ(遅延時間)

発音後、ポルタメントを開始するまでの遅延時間を設定する。値は絶対音長指定で行い、その有効範囲は0-32767。

・hold time or port.time:持続時間または変化時間

負値を与えた場合は、ポルタメントにおける音程変化を行ったあと最終的に、どのくらいの時間これを持続するかの設定を行う(hold time)。また正値を与えた場合はポルタメントにおける音程変化を絶対音長でどれくらいかけるかの設定を行う(port.time)。

それぞれ有効範囲はhold time:-32768～0,port.time:1～32767。

なお、[PUSH_PORTAMENT]と[PORTAMENT]はまったく同一機能。

また、[PUSH_PORTAMENT]と[PULL_PORTAMENT]の違いは基本MMLでいうところの(～)のポルタメントと[～]のポルタメントの違いがある。なお、[PUSH_PORTAMENT]が基本MML(～)に相当し、[PULL_PORTAMENT]が基本MML[～]によるポルタメントに対応する。

例

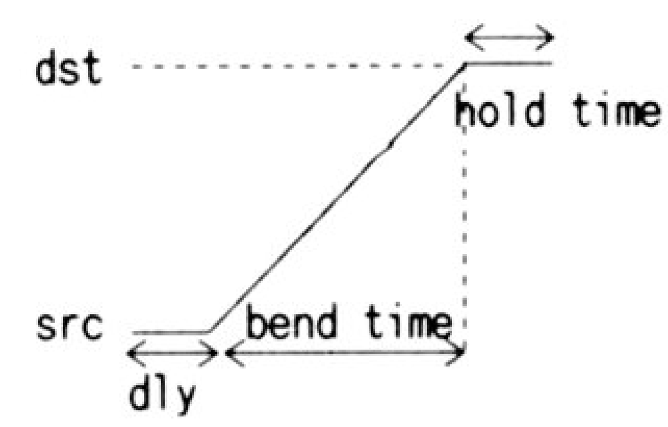
[PORTAMENT o4c,o4e,48,46,120,8,24]

(オクターブ4のcからeに向けてポルタメントを開始する。ポルタメントノートはステップタイムは48、ゲートタイムは46、ベロシティ120で発音する。発音後、絶対音長8の遅延時間の後、絶対音長24かけて目的のノートへポルタメントする。残った時間、絶対音長14(=48-8-24)の間、変化先音程であるオクターブ4のeを持続する)

[PULL_PORTAMENT o4c,o4e,48,46,120,8,-24]

(オクターブ4のcからeに向けてポルタメントを開始する。ポルタメントノートはステップタイムは48、ゲートタイムは46、ベロシティ120で発音する。発音後、絶対音長8の遅延時間の後、絶対音長16(=48-8-24)かけて目的のノートへポルタメントする。絶対音長24の間、変化先音程であるオクターブ4のeを持続する)

FM ADPCM MIDI　オートポルタメント
[AUTO_PORTAMENT sw,dly,hold time(or port.time)]



・sw:0,1または　OFF,ON
・dly:0-32767
・hold time:-32768～0　または　port.time:1～32767

オートポルタメントを制御する。有効にするとある音から次に発音する音符までの音程を断続的に変化させる。

・sw:スイッチ

オートポルタメントを有効にするか無効にするかを設定する。0がOFF,1がONに対応する。またはOFF,ONといった文字列を与えることも可能。

・dly:ディレイ(遅延時間)

オートポルタメントを開始するまでのチェンジ間を絶対音長で与える。有効範囲は0～32767。

・hold.time or port.time:持続時間または変化時間

負値を与えた場合は、オートポルタメントにおける音程変化を行ったあと最終的に、どのくらいの時間これを持続するかの設定を行う(hold time)。また正値を与えた場合はオートポルタメントにおける音程変化を絶対音長でどれくらいかけるかの設定を行う(port.time)。

それぞれ有効範囲はhold time:-32768～0,port.time:1～32767。

※MIDIでオートポルタメントを実行すると暗黙のうちに@J1相当のタイモードに変更される。

FM ADPCM MIDI　オートポルタメントスイッチ
[AUTO_PORTAMENT.SWITCH sw]

・sw:0,1　または OFF,ON

オートポルタメントの動作のON,OFFを行う。swは0がOFF,1がONに対応する。また、文字列 ON,OFFを直接与えることもできる。

例

[AUTO_PORTAMENT.SWITCH 0]　[AUTO_PORTAMENT.SWITCH OFF]
(スイッチオフ)
[AUTO_PORTAMENT.SWITCH 1]　[AUTO_PORTAMENT.SWITCH ON]
(スイッチオン)

■アフタータッチシーケンス

FM ADPCM MIDI　アフタータッチシーケンス設定
[AFTERTOUCH.LEVEL mode, v1,v2,…,v8]

・mode:OFF,1/8,OPTIONAL,ON
・v1～v8:0-127

アフタータッチシーケンスにおけるアフタータッチ値v1～v8の設定を行う。基本MMLの@Zとほぼ同等の機能だが mode が設定できるのが特徴。modeとはアフタータッチシーケンスの動作方式を設定するもの。以下にmodeと対応する動作を示す。

・mode:ON

アフタータッチシーケンスを以前の設定で再開する。

・mode:OFF

アフタータッチシーケンスを停止する。

・mode:1/8

発音する音符の音長の1/8単位ごとでv1からv8のアフタータッチを次々に設定していく。ちょうど基本MMLが提供するアフタータッチシーケンスの機能と同等。

・mode:OPTIONAL
発音する音符の音長とは無関係に、任意の時間区間ごとでv1からv8のアフタータッチを次々に設定していく。時間区間のシーケンスは後述の[AFTERTOUCH.DELAY]で行う。
modeは省略すると、それまでの設定を保持したままとなる。)
v1～v8のアフタータッチ値は任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能。たとえば、

[AFTERTOUCH.LEVEL optional 127,,-10,+5]

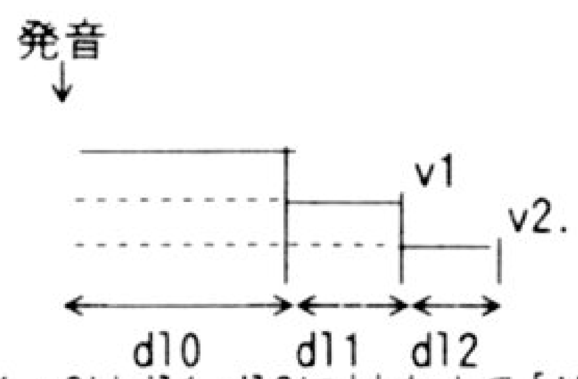
といった記述ができる。省略した部分は前の値を持続するということになり、その間はアフタータッチの情報は送信されない。
相対指定時にオーバーフロー/アンダーフローを起こした値は有効範囲内の最大値,最小値に修正される。

例

[AFTERTOUCH.LEVEL 1/8 0,0,10,20,30]

modeを1/8とし、v1=0,v2=0,v3=10,v4=20,v5=30と設定した。v6,v7,v8は省略(未設定)。

FM ADPCM MIDI　アフタータッチシーケンスディレイ
[AFTERTOUCH.DELAY dl0,dl1,…,dl8]



(v1,v2はdl1,dl2に対応する[AFTERTOUCH.LEVEL]のアフタータッチ値)

・dl0:0～32767
・dl1～dl8:1～65535
アフタータッチシーケンスの動作モードのひとつである「OPTIONAL」モードでのアフタータッチ出力タイミングを決定する区間時間の設定を行う。
dl0は最初のアフタータッチがかかるまでのディレイ(遅延時間)に相当し、絶対音長値で指定する。有効範囲は0～32767。初期値は0。
dl1～dl8は次のアフタータッチ出力までの区間時間で、これも絶対音長値で指定する。有効範囲は1～65535。初期値はなし。
dl0～dl8の任意のものを省略可能だが、dl1～dl8は初期値を持たないので注意。

例

[AFTERTOUCH.DELAY 10,24,12,12,12,28,28,12,12]

dl0=10,dl1=24,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28,dl7=12,dl8=12が設定される。上を設定したあとで、たとえば、

[AFTERTOUCH.DELAY 22,33,,,,,,,44]

とした場合、省略したパラメータについては前の値が保存され、指定されたパラメータのみ書き変わるので、設定は、

dl0=<u>22</u>,dl1=<u>33</u>,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28,dl7=12,dl8=<u>44</u>

のようになる。

FM ADPCM MIDI　アフタータッチシーケンススイッチ
[AFTERTOUCH.SWITCH mode]

・mode:-1～2　または　OFF,1/8,OPTIONAL,ON
アフタータッチの動作制御を行う。modeは[AFTERTOUCH.LEVEL]のパラメータのmodeと同等。modeは数値で与えることもでき、このときは、

0=OFF,-1=1/8,1=OPTIONAL,2=ON

に対応する。

例

[AFTERTOUCH.SWITCH OPTIONAL]
(OPTIONALモードでONにする)

[AFTERTOUCH.SWITCH 2]
(以前のモード(1/8かOPTIONALのいずれか)でONにする)

FM ADPCM MIDI　アフタータッチシーケンス同期モード設定
[AFTERTOUCH.SYNC m,c]

・m:0=非同期　1=同期　　または　OFF,ON
・c:1～65535
アフタータッチシーケンスの初期化(リセット)タイミングの設定を行う。
mは同期モードの設定値。0は非同期,1は同期に対応する。mをON,OFFといった文字列にて与えることも可能で、この場合、文字列OFF,ONは、それぞれ非同期、同期の設定に対応する。
「同期」に設定した場合は新しく発音するたびに、アフタータッチシーケンスを最初から行う。
「非同期」にした場合は発音と無関係にアフタータッチシーケンスを続行する。
数値指定では0,1以外に、以下のように2進数で設定すればより高度な同期化設定が行える。[]は初期設定を表す。

第0ビット	発音と同時にリセット	0:しない[1]:する
第1ビット	タイでもリセット	[0]:しない1:する
第2ビット	休符でもリセット	[0]:NO　1:YES

上からわかるように初期値は1となっている。
cは同期ディレイにあたり、実際にはmで指定した条件がc回成立すると波形が初期化されることになる。cの初期値は1。

例

[AFTERTOUCH.SYNC %011](発音と同時、タイ指定においても初期化する)

■ビブラート(ピッチモジュレーション)

FM ADPCM MIDI　ビブラート振幅
[VIBRATO.DEPTH mode,dp1,dp2,…,dp8]

・mode:OFF,1/8,OPTIONAL,ON
・dp1～dp8:-32768～+32767
ビブラートの振幅の設定をdp1～dp8において行う。基本MMLの@Mとほぼ同等の機能だが mode が設定できるのが特徴。modeはビブラートの動作方式を設定するパラメータ。
以下にmodeと対応する動作を示す。

・mode:ON
ビブラートを以前の設定で再開する。

・mode:OFF
ビブラートを停止する。

・mode:1/8
発音する音符の音長の1/8単位ごとでdp1からdp8のビブラート振幅を次々に変化していく。

・mode:OPTIONAL
発音する音符の音長とは無関係に、任意の時間区間ごとでdp1からdp8のビブラート振幅を次々に設定していく。時間区間のシーケンスは後述の[VIBRATO.DELAY]で行う。
modeは省略するとスイッチはそれまでの設定を保持したままとなる。
dp1～dp8のビブラート振幅は任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能。たとえば、

[VIBRATO.DEPTH optional 64,,-10,+5]

といった記述ができる。省略した部分は前の値を持続するということになり、その間は振幅の変更は行われない。
相対指定時にオーバーフローを起こした値は有効範囲内の最大値,最小値に修正される。

例

[VIBRATO.DEPTH 1/8 0,0,10,20,30]
modeを1/8とし、dp1=0,dp2=0,dp3=10,dp4=20,dp5=30と設定。
dp6,dp7,dp8は省略(未設定)。

FM ADPCM MIDI　ビブラート振幅増加
[VIBRATO.DEEPEN sw,spd,dpn,rpt]

・sw:OFF,ON
・spd:1～65535
・dpn:-32768～+32767
・rpt:0～32767
ビブラートの振幅の増加/減少を設定する。

・sw:OFF,ON
ビブラートの振幅増減機能を有効にするか無効にするかを設定する。
OFFで無効に、ONで有効に設定される。

・spd:増減速度
ビブラートの振幅の増減速度を設定する。1がもっとも速く、65535がもっとも遅い。値は絶対音長に相当し、有効範囲は1～65535。

・dpn:振幅増加量
振幅の増減量を設定する。負値は振幅減少、正値は振幅増加に対応する。
値の有効範囲は-32768～+32767。絶対値が大きければ大きいほど、増減は急激となる。

・rpt:増減回数
振幅はspdの速度でdpn幅増減するが、この処理を何回繰り返すかを設定する。
rpt=0とすると振幅が初期化されるまで無限に増減を繰り返す設定になる。値の有効範囲は1～32767。

例

[VIBRATO.DEEPEN ON,6,+8,10]

（増減スピード6で10回振幅を8増加させる）

FM ADPCM MIDI　ビブラートスピード(周期)
[VIBRATO.SPEED sp1,sp2,…,sp8]

・sp1～sp8:1～32767

ビブラートの振動速度を設定する。基本MMLの@Sとほぼ同等の機能を持つが最大８つの振幅変化に対応した振動速度変化の設定が行えるのが特長。設定範囲は1≦sp1～sp8≦32767。

実際の波形にどう影響するかは拡張MML[VIBRATO.WAVEFORM]の項で解説するが、値が小さいほど高速に、値が大きいほど低速に音が振動する。

sp1～sp8の振動速度は任意の位置で省略可能だが、初期値を持たないので注意。

例

[VIBRATO.SPEED 10,8,20]

(sp1=10,sp2=8,sp3=20,sp4～sp8は省略(未設定))

FM ADPCM MIDI　ビブラートディレイ
[VIBRATO.DELAY dl0,dl1,…,dl8]

・dl0:0～32767

・dl1～dl8:1～65535

ビブラートの動作モードのひとつである「OPTIONAL」モードにおけるビブラートの振幅切り換えタイミングを決定する区間時間の設定を行う。基本MML @Hに相当する機能。

dl0は最初のビブラートがかかるまでのディレイ(遅延時間)に相当し、絶対音長値で指定する。有効範囲は0～32767。初期値は０。

dl1～dl8は次の振幅切り換えまでの区間時間で、これも絶対音長値で指定する。有効範囲は1～65535。初期値はなし。

dl0～dl8の任意のものを省略可能だが、dl1～dl8は初期値を持たないので注意。

例

[VIBRATO.DELAY 10,24,12,12,12,28,28,12,12]

dl0=10,dl1=24,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28,dl7=12,dl8=12が設定される。

FM ADPCM MIDI　ビブラート波形選択
[VIBRATO.WAVEFORM w]

・w:0-32767

ビブラート波形を選択する。基本MMLのSと同等の機能。

w=0～4はプリセット波形に割り当てられている。

・w=0:鋸歯波
・w=1:矩形波
・w=2:三角波
・w=3:ワンショット鋸歯波
・w=4:ランダム波

またはプリセット波形は、

SAW	(w=0:鋸歯波に対応)
SQUARE	(w=1:矩形波に対応)
TRIANGLE	(w=2:三角波に対応)
S.SAW	(w=3:ワンショット鋸歯波に対応)
RANDOM	(w=4:ランダム波に対応)

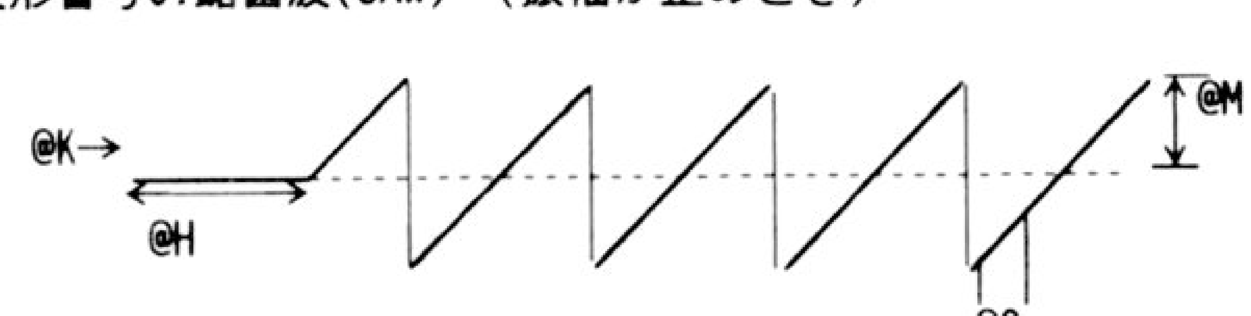
という文字パラメータを用いて指定することもできる。

初期値はw=2の三角波。w=5～7はリザーブ。w=8～32767はユーザー定義波形となる。ユーザー波形の登録についてはMEASURE3 zm_set_wave_from1(),MEASURE4 「.WAVE_FORM」参照。

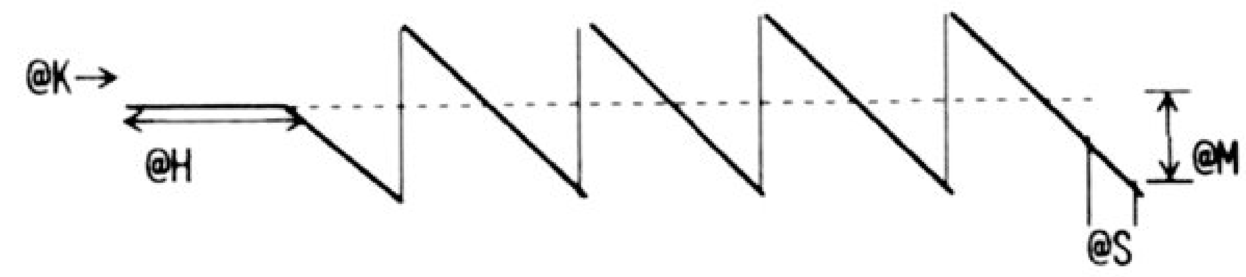
以下に波形と各パラメータとの関係を示す。

@M:振幅	[VIBRATO.DEPTH]
@H:ディレイ	[VIBRATO.DELAY]
@K:ディチューン	[VIBRATO.DETUNE]
@S:速度	[VIBRATO.SPEED]

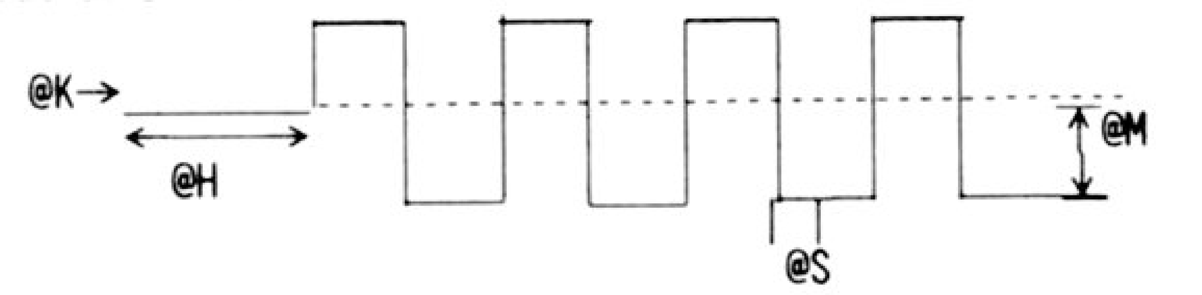
波形番号0:鋸歯波(SAW)（振幅が正のとき）



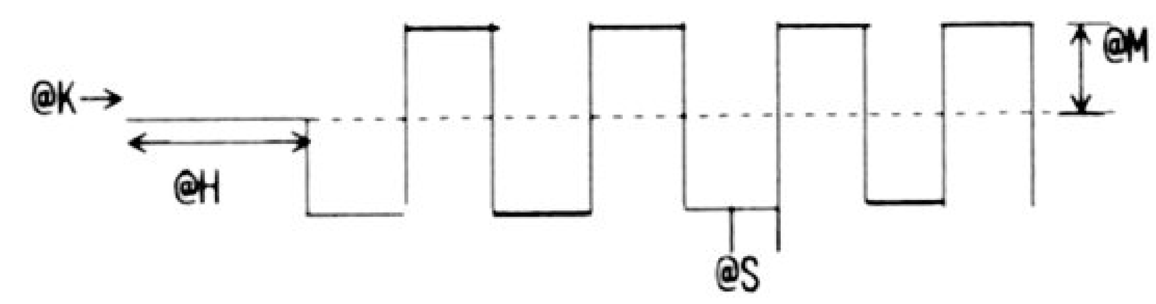
波形番号0:鋸歯波（振幅が負のとき）



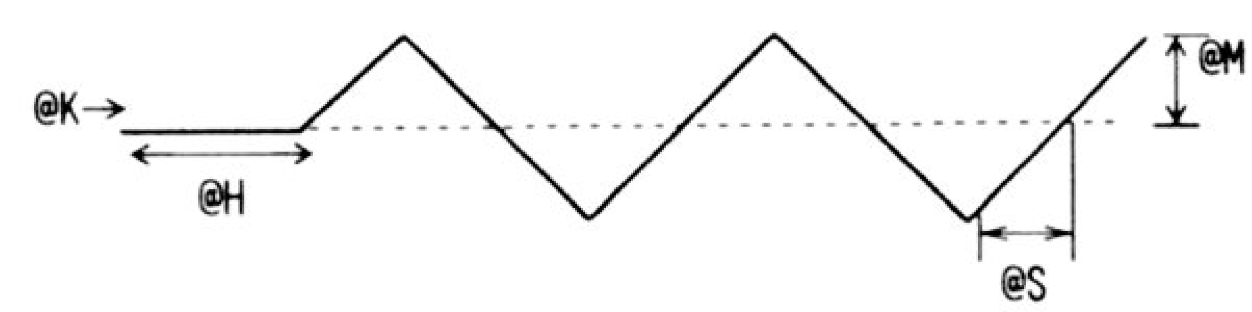
波形番号1:矩形波（振幅が正のとき）



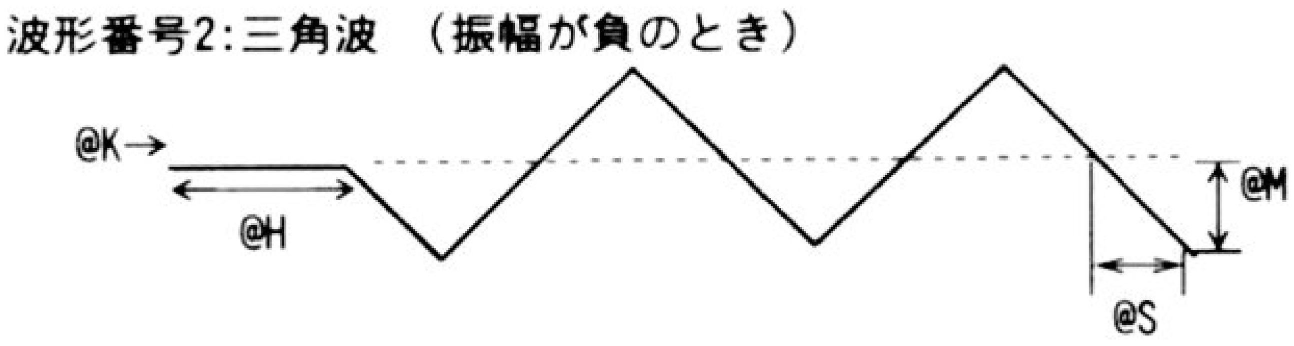
波形番号1:矩形波（振幅が負のとき）



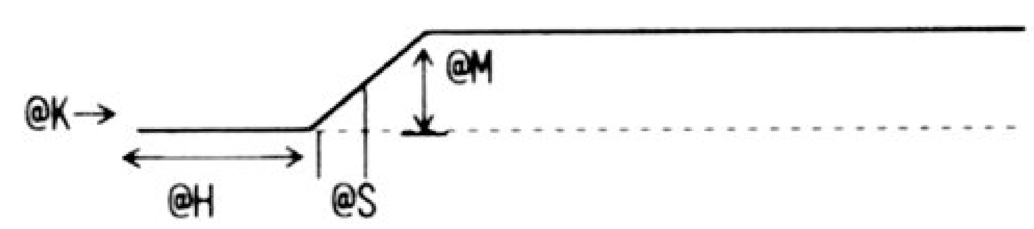
波形番号2:三角波（振幅が正のとき）



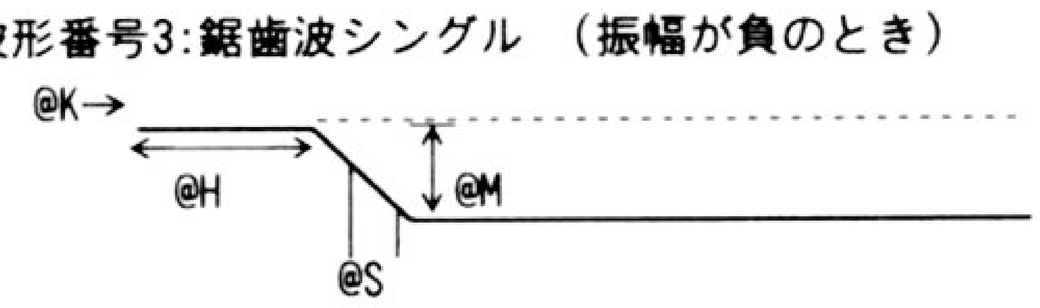
波形番号2:三角波（振幅が負のとき）



波形番号3:鋸歯波シングル（振幅が正のとき）

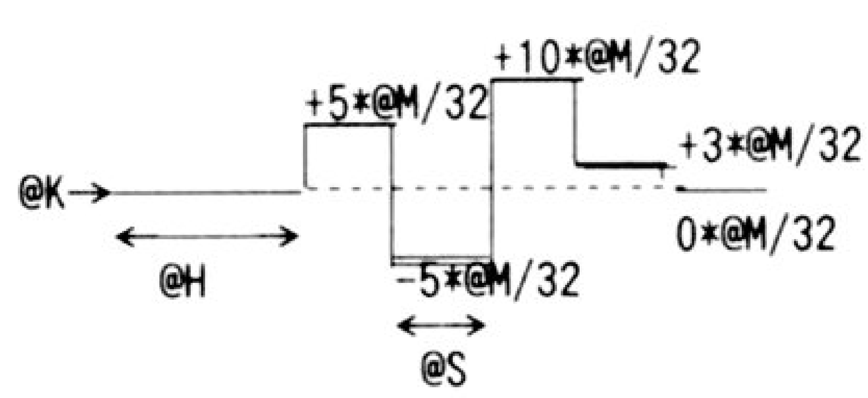


波形番号3:鋸歯波シングル（振幅が負のとき）



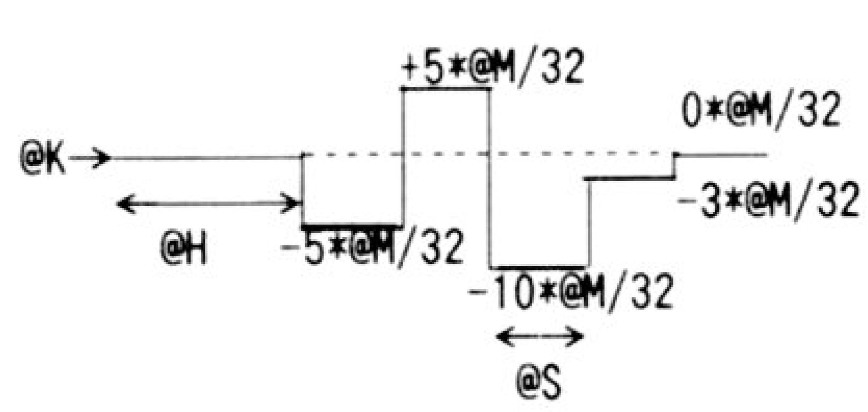
波形番号4:ランダム波（振幅が正のとき）
波形番号8～32767:ユーザー波形（振幅が正のとき）

波形データ列が+5,−5,+10,+3,0の場合・・・



波形番号4:ランダム波（振幅が負のとき）
波形番号8～32767:ユーザー波形（振幅が負のとき）

波形データ列が+5,−5,+10,+3,0の場合・・・



FM ADPCM MIDI　ビブラートスイッチ
[VIBRATO.SWITCH mode]

・mode:-1～2　または　OFF,1/8,OPTIONAL,ON

ビブラートの動作制御を行う。modeは[VIBRATO.DEPTH]のパラメータのmodeと同等。modeは数値で与えることもでき、このときは、

0=OFF,-1=1/8,1=OPTIONAL,2=ON

に対応する。

例

[VIBRATO.SWITCH 1/8]

(1/8モードでONにする)

[VIBRATO.SWITCH 2]

(以前のモード(1/8かOPTIONALのいずれか)でONにする)

FM ADPCM MIDI　ビブラート同期モード
[VIBRATO.SYNC m,c]

・m:0=非同期　1=同期　　または　OFF,ON

・c:1〜65535

ビブラート波形の初期化(リセット)タイミングの設定を行う。基本MML Hに相当する機能を持つ。

mは同期モードの設定値。0は非同期,1は同期に対応する。mをON,OFFといった文字列にて与えることも可能で、この場合、文字列OFF,ONは、それぞれ非同期、同期の設定に対応する。

「同期」に設定した場合は新しく発音するたびにビブラート波形の初期化を行う。

「非同期」にした場合は発音と無関係にビブラートを続行する。

数値指定では0,1以外に、以下のように2進数で設定すればより高度な同期設定が行える。[]は初期設定を表す。

第0ビット	発音と同時にリセット	0:しない[1]:する
第1ビット	タイでもリセット	[0]:しない1:する
第2ビット	休符でもリセット	[0]:NO　1:YES
第3ビット	波形はキーオンごとに進行する	[0]:NO　1:YES
第4ビット	波形は休符でも進行する	[0]:NO　1:YES

上からわかるように初期値は1となっている。

cは同期ディレイにあたり、実際にはmで指定した条件がc回成立すると波形が初期化されることになる。cの初期値は1。

例

[VIBRATO.SYNC %011]

(発音と同時、タイ指定においても初期化する)

MIDI　ビブラートモード
[VIBRATO.MODE mode]

・mode:0〜2　または NORMAL,FM,MIDI

MIDIにはビブラートモードが2つあり本命令はこのモード選択をするもので、基本MML Mに相当する。ビブラートについての詳解は基本MML Mの項を参照のこと。

・mode=0,NORMAL

ノーマルモードの設定。

「[VIBRATO.DEPTH]で指定できる振幅の範囲は0≦@M≦127。

・mode=1,FM

拡張モード1の設定。[VIBRATO.DEPTH]で指定される振幅は半音=64として作用する。[VIBRATO.DEPTH]で指定できる振幅の範囲は−768≦@M≦768。

・mode=2,MIDI

拡張モード2の設定。設定された振幅の値はそのまま14ビットのMIDIのピッチベンダー直値に該当する。(ベンドレンジ=12のとき半音≒683)。

[VIBRATO.DEPTH]で指定できる振幅の範囲は-8192≦@M≦8191。

※注意

本命令を実行するとそれまで設定されていた振幅([VIBRATO.DEPTH]の値)は無効となる。

例

[VIBRATO.MODE NORMAL]　　(ビブラートモードを通常モードに設定する)

■ARCC(Assignable Realtime Control Change)

FM ADPCM MIDI　ARCC振幅
[ARCCn.LEVEL mode,lv1,lv2,…,lv8]
[ARCCn.DEPTH mode,lv1,lv2,…,lv8]

・n:1-4

・mode:OFF,1/8,OPTIONAL,ON

・lv1〜lv8:-128〜+127

ARCCの振幅設定を行う。基本MMLの@Aほぼの機能だが@AはARCC1に対しての設定しか行えないのに対して、本命令は各トラック独立動作させることのできる4つのARCCコントローラーに対して個別にパラメータを設定できる。

nはARCC番号で設定範囲は1〜4。

modeはARCCの動作方式を設定するパラメータ。以下にmodeと対応する動作を示す。

・mode:ON

ARCCを以前の設定で再開する。

・mode:OFF

ARCCを停止する。

・mode:1/8

発音する音符の音長の1/8単位ごとでlv1からlv8のARCC値を次々に設定していく。

・mode:OPTIONAL

発音する音符の音長とは無関係に、任意の時間区間ごとでlv1からlv8のARCC振幅を次々に設定していく。時間区間のシーケンスは後述の[ARCCn.DELAY]で行う。

modeは省略するとスイッチはそれまでの設定を保持したままとなる。

lv1〜lv8のARCC振幅は任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能。たとえば、

[ARCCn.LEVEL optional 64,,-10,+5]

といった記述ができる。省略した部分は前の値を持続するということになり、その間は振幅の変更は行われない。

相対指定時にオーバーフローを起こした値は有効範囲内の最大値,最小値に修正される。

例

[ARCC1.LEVEL 1/8 0,0,10,20,30]

modeを1/8とし、lv1=0,lv2=0,lv3=10,lv4=20,lv5=30と設定。

lv6,lv7,lv8は省略(未設定)。

FM ADPCM MIDI　ARCC振幅増加
[ARCCn.DEEPEN sw,spd,dpn,rpt]

・n:1-4

・sw:OFF,ON

・spd:1〜65535

・dpn:-32768〜+32767

・rpt:0〜32767

ARCCの振幅の増加/減少を設定する。

・sw:OFF,ON

ARCCの振幅増減機能を有効にするか無効にするかを設定する。

OFFで無効に、ONで有効に設定される。

・spd:増減速度

ARCCの振幅の増減速度を設定する。1がもっとも速く、65535がもっとも遅い。値は絶対音長に相当し、有効範囲は1〜65535。

・dpn:振幅増加量

振幅の増減量を設定する。負値は振幅減少、正値は振幅増加に対応する。

値の有効範囲は-32768〜+32767。絶対値が大きければ大きいほど、増減は急激となる。

・rpt:増減回数

振幅はspdの速度でdpn幅増減するが、この処理を何回繰り返すかを設定する。rpt=0とすると振幅が初期化されるまで無限に増減を繰り返す設定になる。値の有効範囲は1〜32767。

例

[ARCC1.DEEPEN ON,6,+8,10]

(ARCC1を増減スピード6で10回振幅を8増加させる)

FM ADPCM MIDI　ARCCコントロール設定
[ARCCn.CONTROL ctrl]

・n=:1-4

・ctrl:0-127

ARCCの制御対象コントロールを設定する。基本MMLの@Cほぼの機能だが@CはARCC1に対しての設定しか行えないのに対して、本命令は各トラック独立動作させることのできる4つのARCCコントローラに対して個別にパラメータを設定できる。

nはARCC番号で設定範囲は1〜4。

ctrlは制御対象コントロール番号。MIDIパートならばARCCで制御したいコントロールチェンジ番号を与えることになる。MIDIパートのコントロールは以下の文字列を設定することでも行える。

0=BANK_MSB 1=VIBRATO 2=CTRL2 3=CTRL3 4=CTRL4 5=PORTAMENT_TIME
6=DATA_ENTRY_MSB 7=VOLUME 8=CTRL8 9=CTRL9 10=PANPOT
11=EXPRESSION 12=CTRL12 13=CTRL13 14=CTRL14 15=CTRL15
16=CTRL16 17=CTRL17 18=CTRL18 19=CTRL19 20=CTRL20 21=CTRL21
22=CTRL22 23=CTRL23 24=CTRL24 25=CTRL25 26=CTRL26 27=CTRL27

28=CTRL28 29=CTRL29 30=CTRL30 31=CTRL31 32=BANK_LSB 33=CTRL33
34=CTRL34 35=CTRL35 36=CTRL36 37=DATA_ENTRY_LSB 38=CTRL31
39=CTRL39 40=CTRL40 41=CTRL41 42=CTRL42 43=CTRL43 44=CTRL44
45=CTRL45 46=CTRL46 47=CTRL47 48=CTRL48 49=CTRL49 50=CTRL50
51=CTRL51 52=CTRL52 53=CTRL53 54=CTRL54 55=CTRL55 56=CTRL56
57=CTRL57 58=CTRL58 59=CTRL59 60=CTRL60 61=CTRL61 62=CTRL62
63=CTRL63 64=DAMPER 65=PORTAMENT 66=SOSTENUTO 67=SOFT
68=CTRL68 69=FREEZE 70=CTRL70 71=CTRL71 72=CTRL72 73=CTRL73
74=CTRL74 75=CTRL75 76=CTRL76 77=CTRL77 78=CTRL78 79=CTRL79
80=CTRL80 81=CTRL81 82=CTRL82 83=CTRL83 84=PORTAMENT_CONTROL
85=CTRL85 86=CTRL86 87=CTRL87 88=CTRL88 89=CTRL89 90=CTRL90
91=REVERB 92=TREMOLO 93=CHORUS 94=DELAY 95=PHASER
96=DATA_INCREMENT 97=DATA_DECREMENT 98=NRPN_LSB 99=NRPN_MSB
100=RPN_LSB 101=RPN_MSB 102=CTRL102 103=CTRL103 104=CTRL104
105=CTRL105 106=CTRL106 107=CTRL107 108=CTRL108 109=CTRL109
110=CTRL110 111=CTRL111 112=CTRL112 113=CTRL113 114=CTRL114
115=CTRL115 116=CTRL116 117=CTRL117 118=CTRL118 119=CTRL119
120=ALL_SOUND_OFF 121=RESET_ALL_CONTROLLERS 122=LOCAL
123=ALL_NOTES_OFF 124=OMNI_OFF 125=OMNI_ON 126=MONO 127=POLY

例

[ARCC1.CONTROL EXPRESSION]
[ARCC1.CONTROL 11]
(上記2つは同義)

FM音源パートのctrlは%0000〜%1111で操作対象オペレータの登録となる。それぞれビット0〜3までがOPMのオペレータ1〜4に対応する。また以下の文字列でコントロール設定を行えばFM音源の特殊パラメータに対してのARCCが行える。()内の数値はモジュレート有効範囲を表す。たとえばFM.AMSに対して-128〜127の振幅でモジュレートしても実際に有効となる値は0-3のみ。

$80=FM.PANPOT (0-127) *1
$81=FM.PMS (0-7)
$82=FM.AMS (0-3)
$84=FM.AMD (0-127)
$85=FM.PMD (0-127)
$86=FM.LFO (0-127) *2
$87=FM.NOISE(0-63)

*1 パラメータ的には0-127で行われるが実際の効果は左、中央、右の3段階にしか鳴らない
*2 FM音源のレジスタには0-127の2倍の値(0-254)が書き込まれる

例

[ARCC1.CONTROL FM.AMD]
[ARCC1.CONTROL $84]
(上記2つは同義)

AD PCMパートのctrlはMIDIに準拠するが、現時点でサポートされているのは、
7=VOLUME 10=PANPOT 11=EXPRESSION
の3つのみ。

FM ADPCM MIDI ARCC波形の原点設定
[ARCCn.ORIGIN org]

・org:0-127

ARCC波形の基準値を設定する。この値を振幅の上限値として[ARCCn.DEPTH]([ARCCn.LEVEL])で設定した振幅で振動する。この値は[ARCCn.WAVEFORM]の図解中の@Cに相当する。初期値は127。

例

[ARCC1.ORIGIN 64] (ARCC1の波形の基準値を64に設定する)

FM ADPCM MIDI ARCC波形の初期化値の設定
[ARCCn.RESET rst]

・rst:0-127

ARCC波形の初期化値を設定する。後述の[ARCCn.MODE]の設定によって多少作用が異なる。初期値は0。

・[ARCCn.MODE]がNORMALのとき

ARCC開始時にこの値でコントローラが初期化される(ただし[ARCCn.DELAY]のdl≠0のとき)。またARCC解除時にもこの値でコントローラが初期化される。

・[ARCCn.MODE]がSPECIAL(ENHANCED)のとき

ARCC解除時にもこの値でコントローラが初期化される。

例

[ARCC1.RESET 0] (ARCC1の初期化値を0に設定する)

FM ADPCM MIDI ARCCスピード(周期)
[ARCCn.SPEED sp1,sp2,…,sp8]

・n:1-4
・sp1〜sp8:1〜32767

ARCCの振動速度を設定する。基本MMLの@Sとほぼ同等の機能を持つが最大8つの振幅変化に対応した振動速度変化の設定が行えるのが特長。nは設定先ARCC番号で設定範囲は1〜4。

sp1〜8は速度値で設定範囲は1≦sp1〜sp8≦32767。

実際の波形にどう影響するかは拡張MML[ARCCn.WAVEFORM]の項で解説するが、値が小さいほど高速に、値が大きいほど低速に音が振動する。

sp1〜sp8の振動速度は任意の位置で省略可能だが、初期値を持たないので注意。

例

[ARCC1.SPEED 10,8,20]
(ARCC1の速度をsp1=10,sp2=8,sp3=20と設定,sp4〜sp8は省略(未設定))

FM ADPCM MIDI ARCCディレイ
[ARCCn.DELAY dl0,dl1,…,dl8]

・n:1-4
・dl0:0〜32767
・dl1〜8:1〜65535

ARCCの動作モードのひとつである「OPTIONAL」モードにおけるARCCの振幅切り換えタイミングを決定する区間時間の設定を行う。基本MML @Hに相当する機能。

nは設定先ARCC番号で設定範囲は1〜4。

dl0は最初のARCCが実行されるまでのディレイ(遅延時間)に相当し、絶対音長値で指定する。有効範囲は0〜32767。初期値は0。

dl1〜dl8は次の振幅切り換えまでの区間時間で、これも絶対音長値で指定する。有効範囲は1〜65535。初期値はなし。

dl0〜dl8の任意のものを省略可能だが、dl1〜dl8は初期値を持たないので注意。

例

[ARCC1.DELAY 10,24,12,12,12,28,28,12,12]
(ARCC1のディレイをdl0=10,dl1=24,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28, dl7=12,dl8=12と設定する)

FM ADPCM MIDI ARCC波形
[ARCCn.WAVEFORM w]

・n:1-4
・w:0-32767

ARCC波形を選択する。基本MMLのSと同等の機能。

nは設定先ARCC番号で設定範囲は1〜4。

wの0〜4はプリセット波形に割り当てられている。

・w=0:鋸歯波
・w=1:矩形波
・w=2:三角波
・w=3:ワンショット鋸歯波
・w=4:ランダム波

またはプリセット波形は、

SAW (w=0:鋸歯波に対応)
SQUARE (w=1:矩形波に対応)
TRIANGLE (w=2:三角波に対応)
S.SAW (w=3:ワンショット鋸歯波に対応)
RANDOM (w=4:ランダム波に対応)

という文字パラメータを用いて指定することもできる。

初期値はw=2の三角波。

w=5〜7はリザーブ。w=8〜32767はユーザー定義波形となる。

ユーザー波形の登録についてはMEASURE3 zm_set_wave_form1(),MEASURE4 「.WAVE_FORM」参照。

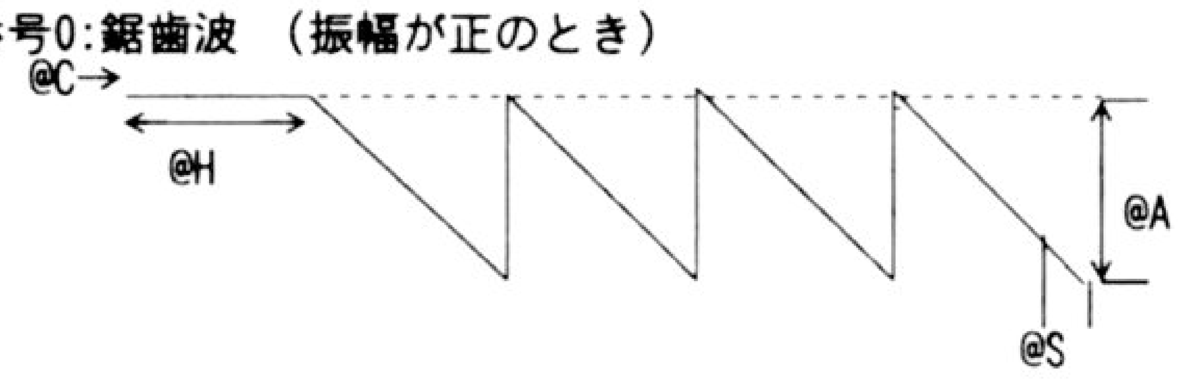
以下に波形と各パラメータとの関係を示す。

@A:振幅 [ARCCn.DEPTH]([ARCCn.LEVEL])
@H:ディレイ [ARCCn.DELAY]
@C:波形基準値※ [ARCCn.ORIGIN]
(ARCCn.CONTROL/@Cの第3パラメータ)
@S:速度 [ARCCn.SPEED]

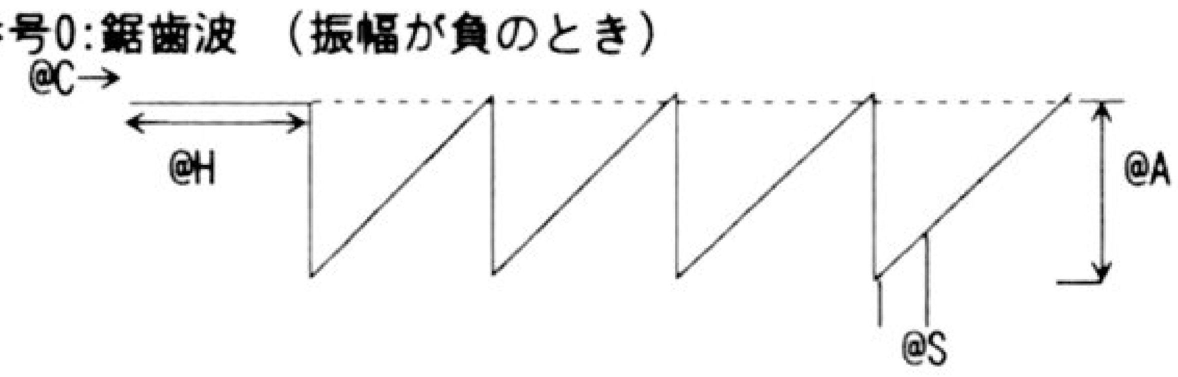
※FM音源パートおよびAD PCMパートでは[ARCCn.CONTROL]を設定していない場合、ARCCはアンプリチュードモジュレーションとして動作する。この場合は上記の@Cで表される波形基準値はその時点での設定音量に相当する。

FM音源パートおよびAD PCMパートでも[ARCCn.CONTROL]を設定している場合は、MIDIパートと同様のARCCとして機能する。

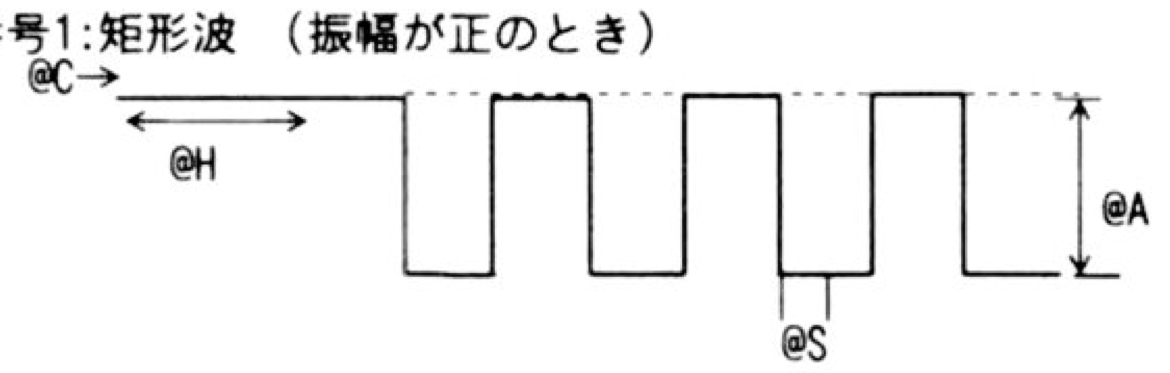
波形番号0:鋸歯波　(振幅が正のとき)



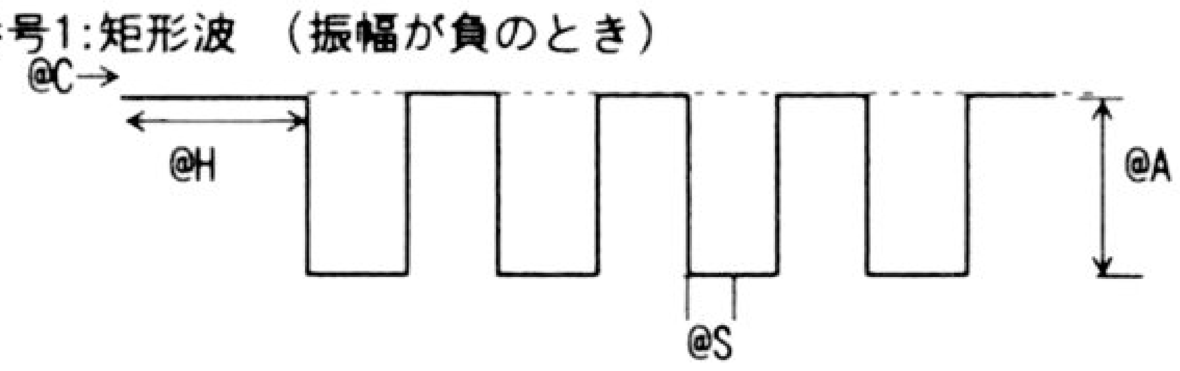
波形番号0:鋸歯波　(振幅が負のとき)



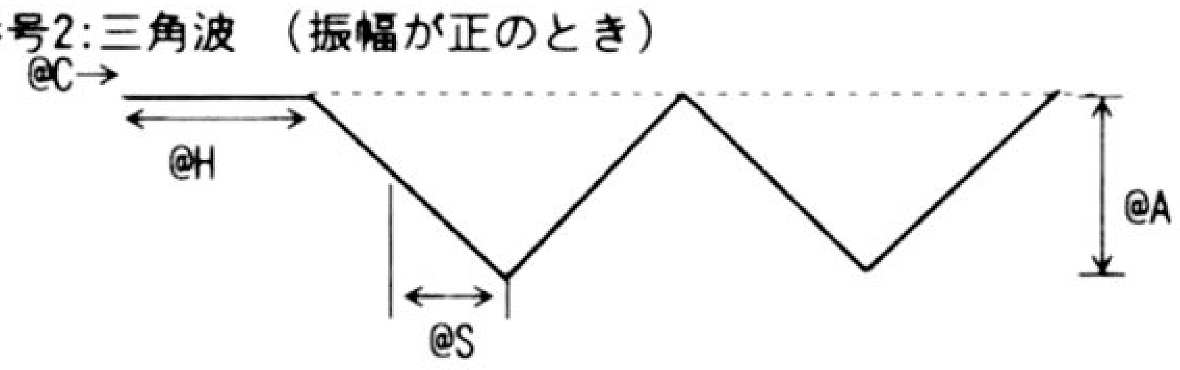
波形番号1:矩形波　(振幅が正のとき)



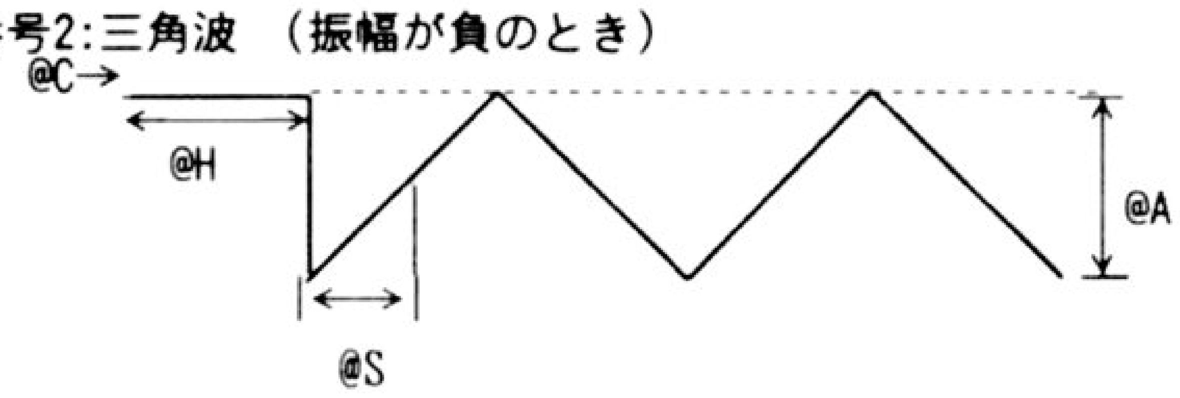
波形番号1:矩形波　(振幅が負のとき)



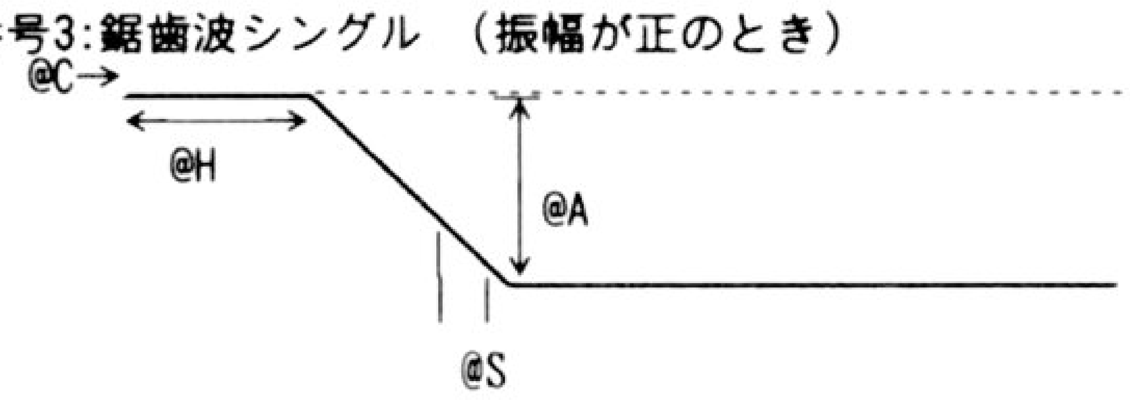
波形番号2:三角波　(振幅が正のとき)



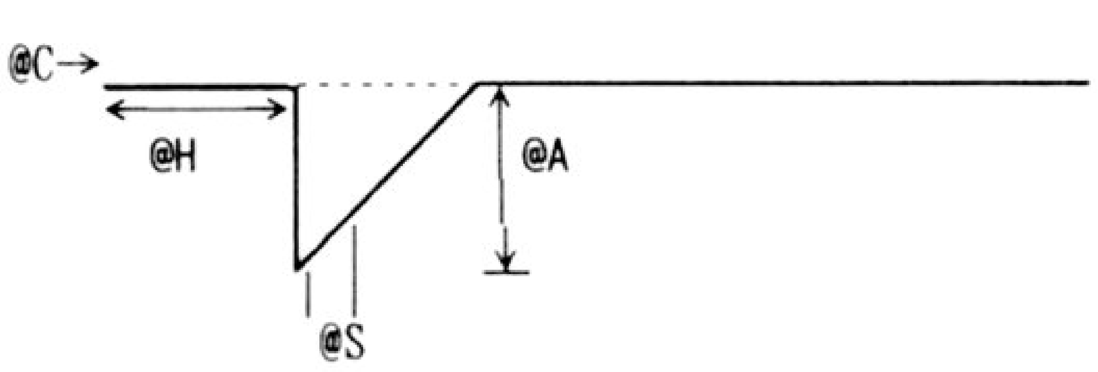
波形番号2:三角波　(振幅が負のとき)



波形番号3:鋸歯波シングル　(振幅が正のとき)

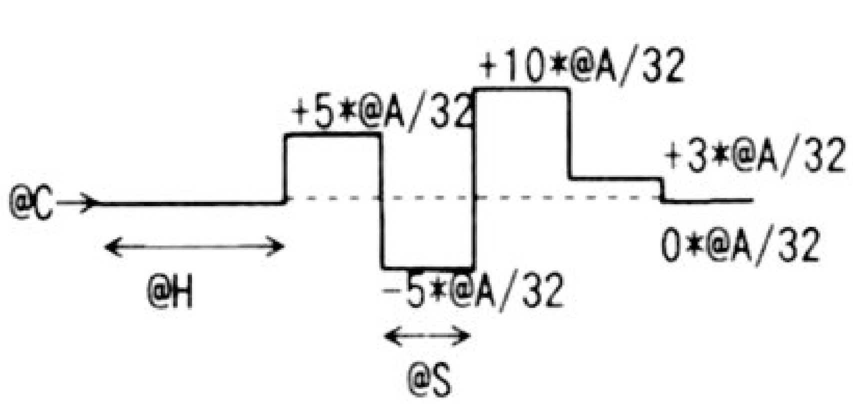


波形番号3:鋸歯波シングル　(振幅が負のとき)



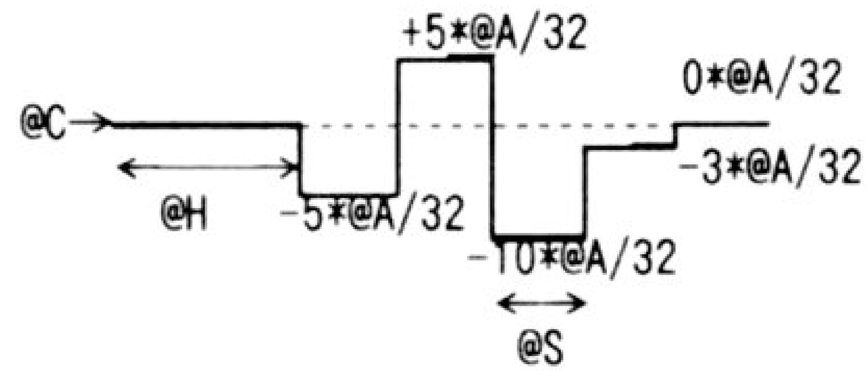
波形番号4:ランダム波　(振幅が正のとき)
波形番号8～32767:ユーザー波形　(振幅が正のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



波形番号4:ランダム波　(振幅が負のとき)
波形番号8～32767:ユーザー波形　(振幅が負のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



FM ADPCM MIDI　ARCCスイッチ
[ARCCn.SWITCH mode]

・n:1-4
・mode:-1～2　または　OFF,1/8,OPTIONAL,ON

ARCCの動作制御を行う。
nは設定先ARCC番号で設定範囲は1～4。
modeは[ARCCn.DEPTH]([ARCCn.LEVEL])のパラメータのmodeと同等。modeは数値で与えることもでき、このときは、

0=OFF,-1=1/8,1=OPTIONAL,2=ON

に対応する。

例

[ARCC2.SWITCH 1/8]
(ARCC2を1/8モードでONにする)
[ARCC4.SWITCH 2]
(ARCC4を以前のモード(1/8かOPTIONALのいずれか)でONにする)

FM ADPCM MIDI　ARCC同期モード
[ARCCn.SYNC m,c]

・n:1-4
・m:0=非同期　1=同期　　または　OFF,ON
・c:1～65535

ARCC波形の初期化(リセット)タイミングの設定を行う。基本MML Hに相当する機能を持つ。
nは設定先ARCC番号で設定範囲は1～4。
mは同期モードの設定値。0は非同期,1は同期に対応する。mをON,OFFといった文字列にて与えることも可能で、この場合、文字列OFF,ONは、それぞれ非同期、同期の設定に対応する。
「同期」に設定した場合は新しく発音するたびにARCC波形の初期化を行う。
「非同期」にした場合は発音と無関係にARCCを続行する。
数値指定では0,1以外に、以下のように2進数で設定すればより高度な同期設定が行える。[]は初期設定を表す。

第0ビット	発音と同時にリセット	0:しない[1]:する
第1ビット	タイでもリセット	[0]:しない1:する
第2ビット	休符でもリセット	[0]:NO　1:YES
第3ビット	波形はキーオンごとに進行する	[0]:NO　1:YES
第4ビット	波形は休符でも進行する	[0]:NO　1:YES

上からわかるように初期値は1となっている。
cは同期ディレイにあたり、実際にはmで指定した条件がc回成立すると波形が初期化されることになる。cの初期値は1。

例

[ARCC2.SYNC %011]
(ARCC2を発音と同時、タイ指定においても初期化する)

FM ADPCM MIDI　ARCCモード
[ARCCn.MODE mode]

・n:1-4
・mode:0～2　または NORMAL,SPECIAL(ENHANCED)

MIDIにはARCCモードが2つあり本命令はこのモード選択をするもので、基本MML Mに相当する。ARCCについての詳解は基本MML Mの項を参照のこと。
nは設定先ARCC番号で設定範囲は1～4。

・mode=0,NORMAL
ノーマルモードの設定。
[ARCCn.DEPTH]([ARCCn.LEVEL])で指定できる振幅の範囲は0≦@A≦127。
・mode=1,SPECIAL
拡張モードの設定。ARCCは選択した波形に沿って実行される。
[ARCCn.DEPTH]([ARCCn.LEVEL])で指定できる振幅の範囲は−128≦@A≦127。

※注意
本命令を実行するとそれまで設定されていた振幅([ARCCn.DEPTH],[ARCCn.LEVEL])は無効となる。

例

[ARCC3.MODE NORMAL]

(ARCC3のARCCモードを通常モードに設定する)

FM ADPCM MIDI　ARCC波形位相制御
[ARCCn.PHASE mode]

・n:1-4
・mode:0～1　または NORMAL,REVERSE(INVERSE)

ARCCの波形の位相を制御する。通常とデフォルト状態ではmode=0(NORMAL)に相当し、[ARCCn.WAVEFORM]の項で図解した波形が実行される。一方mode=1(REVERSE)では[ARCCn.WAVEFORM]の項で図解した波形において@Cを中心に反転した波形が実行される。

例

[ARCC3.PHASE REVERSE]　(ARCC3のARCC波形を逆位相モードにする)

■音量／ベロシティ

FM ADPCM MIDI　音量
[VOLUME v]

・v:0-127　あるいは　-127～+127 (+,-の符号を明記すると相対指定)

音量の設定を行う。基本MML @Vに相当する。vの設定範囲は0≦v≦127。数値の頭に+,-の符号を付記すると相対指定になる。この場合のvの設定範囲は-127≦v≦+127。

例

[VOLUME 42]　(音量42を設定)
[VOLUME -12]　(それまでの音量を12下げる)

FM ADPCM MIDI　相対音量
[@VOLUME v]

・v:-127～127

音量の設定を相対指定で行う。基本MML @Vに相当する。vの設定範囲は-127≦v≦+127。[VOLUME]とは違い先頭に+,-をつけなくても相対指定として見なされる。

例

[@VOLUME 12]　(それまでの音量を12上げる)
[@VOLUME -12]　(それまでの音量を12下げる)

FM ADPCM MIDI　ベロシティ
[VELOCITY u]

・u:-127～+127あるいは -127～+127 (+,-の符号を明記すると相対指定)

ベロシティの設定を行う。基本MML @Uに相当する。uの設定範囲は0≦u≦127。数値の頭に+,-の符号を付記すると相対指定になる。この場合のuの設定範囲は-127≦u≦+127。

例

[VELOCITY 42]　(ベロシティを42に設定)
[VELOCITY -12]　(それまでのベロシティを12下げる)

FM音源パート,AD PCM音源パートでは音量とベロシティの区別はなく同一の効果として扱われる点に注意。たとえば以下に示す設定はまったく同一の効果をもたらす設定となる。

例

[VELOCITY 42]　(ベロシティを42に設定)
[VOLUME 42]　(音量を42に設定)

FM ADPCM MIDI　相対ベロシティ
[@VELOCITY u]

・u:-127～127

ベロシティの設定を相対指定で行う。基本MML @Uに相当する。uの設定範囲は-127≦u≦+127。[VELOCITY]とは違い先頭に+,-をつけなくても相対指定として見なされる。

例

[@VELOCITY 12]　(それまでのベロシティを12上げる)
[@VELOCITY -12]　(それまでのベロシティを12下げる)

■EVS(Enhanced Velocity Sequence)

FM ADPCM MIDI　EVS振幅
[VELOCITY.DEPTH mode,lv1,lv2,…,lv8]
[VELOCITY.LEVEL mode,lv1,lv2,…,lv8]

・mode:OFF,OPTIONAL,ON
・lv1～lv8:-128～+127

EVSとはZ-MUSIC ver.1.0からの機能であるベロシティシーケンスを拡張したもので、発音音符のベロシティをビブラートやARCCのように算術波形やユーザー波形に沿って変化させる機能である。本命令は、このEVSの変化振幅を設定するもの。

modeはEVSの動作方式を設定するパラメータ。以下にmodeと対応する動作を示す。

・mode:ON
　EVSを以前の設定で再開する。
・mode:OFF
　EVSを停止する。
・mode:OPTIONAL
　[VELOCITY.DELAY]で設定された音符数ごとにlv1からlv8のEVS振幅を切り換えていく。

※EVSは音長という概念がないのでビブラートやARCCのような1/8モードはない。

modeは省略するとスイッチはそれまでの設定を保持したままとなる。

lv1～lv8のEVS振幅は任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能。たとえば、

[VELOCITY.LEVEL optional 64,,-10,+5]

といった記述ができる。省略した部分は前の値を持続するということになり、その間は振幅の変更は行われない。

相対指定時にオーバーフローを起こした値は有効範囲内の最大値,最小値に修正される。

例

[VELOCITY.LEVEL 1/8 0,0,10,20,30]
(modeを1/8とし、lv1=0,lv2=0,lv3=10,lv4=20,lv5=30と設定した。lv6,lv7,lv8は省略(未設定))

FM ADPCM MIDI　EVSスピード(周期)
[VELOCITY.SPEED sp1,sp2,…,sp8]

・sp1～sp8:1～32767

EVSの振動速度を設定する。最大8つの振幅変化に対応した振動速度変化の設定を行う。sp1～8は速度値で設定範囲は1≦sp1～sp8≦32767。

実際の波形にどう影響するかは拡張MML[VELOCITY.WAVEFORM]の項で解説するが、値が小さいほど高速に、値が大きいほど低速に音が振動する。

sp1～sp8の振動速度は任意の位置で省略可能だが、初期値を持たないので注意。

例

[VELOCITY.SPEED 10,8,20]
(EVSの速度をsp1=10,sp2=8,sp3=20と設定,sp4～sp8は省略(未設定))

FM ADPCM MIDI　EVSディレイ
[VELOCITY.DELAY dl0,dl1,…,dl8]

・dl0:0～65535
・dl1～8:1～65535

EVSの動作モードのひとつである「OPTIONAL」モードにおけるEVSの振幅切り換えタイミングを決定する区間時間の設定を行う。

dl0は最初のEVSが実行されるまでのディレイ(遅延時間)に相当し、音符個数値で指定する。有効範囲は0～65535。初期値は0。

dl1～dl8は次の振幅切り換えまでの区間時間で、これも音符個数値で指定する。有効範囲は1～65535。初期値はなし。

dl0～dl8の任意のものを省略可能だが、dl1～dl8は初期値を持たないので注意。

例

[VELOCITY.DELAY 10,24,12,12,12,28,28,12,12]
(EVSのディレイをdl0=10,dl1=24,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28,dl7=12,dl8=12と設定する。つまり音符dl=10個分はEVSの影響なく発音し11個目からdl1=24個分最初の振幅値でEVS開始となる。以下同様)

FM ADPCM MIDI　EVS波形
[VELOCITY.WAVEFORM w]

・w:0-32767

EVS波形を選択する。

wの0～4はプリセット波形に割り当てられている。

・w=0:鋸歯波
・w=1:矩形波
・w=2:三角波
・w=3:ワンショット鋸歯波
・w=4:ランダム波

またはプリセット波形は、

SAW　(w=0:鋸歯波に対応)

SQUARE　　　　(w=1:矩形波に対応)
TRIANGLE　　　(w=2:三角波に対応)
S.SAW　　　　　(w=3:ワンショット鋸歯波に対応)
RANDOM　　　　(w=4:ランダム波に対応)

という文字パラメータを用いて指定することもできる。

初期値はw=2の三角波。

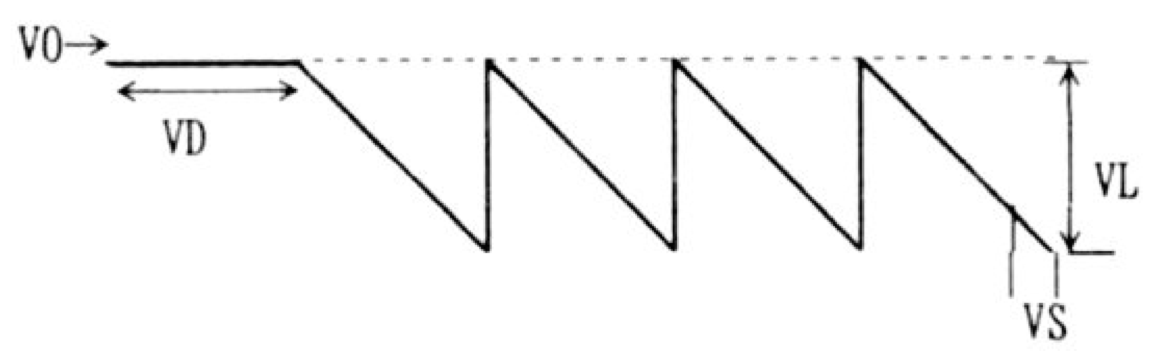
w=5～7はリザーブ。w=8～32767はユーザー定義波形となる。

ユーザー波形の登録についてはMEASURE3 zm_set_wave_from1(),MEASURE4「.WAVE_FORM」参照。

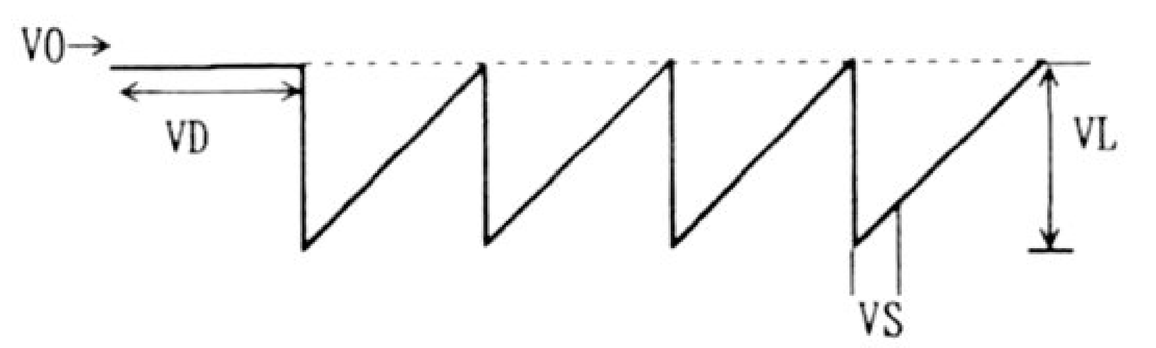
以下に波形と各パラメータとの関係を示す。

VL:振幅　　　[VELOCITY.DEPTH]([VELOCITY.LEVEL])
VD:ディレイ　[VELOCITY.DELAY]
VO:波形基準値　[VELOCITY.ORIGIN]
VS:速度　　　[VELOCITY.SPEED]

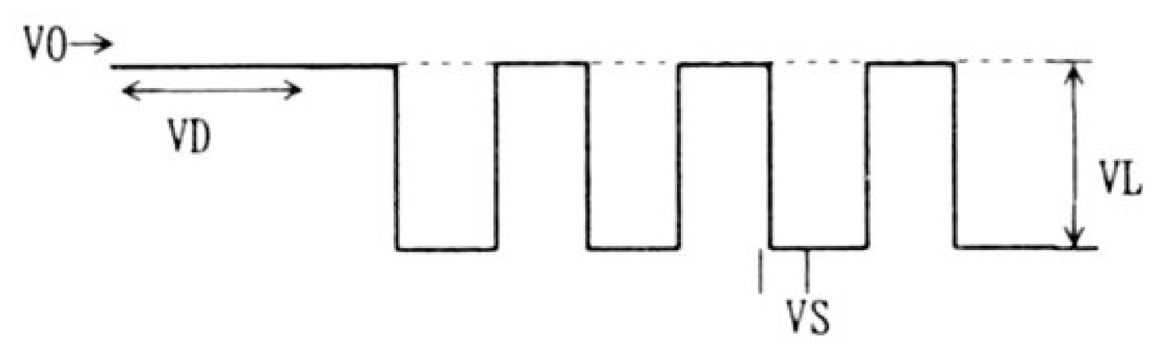
波形番号0:鋸歯波　(振幅が正のとき)



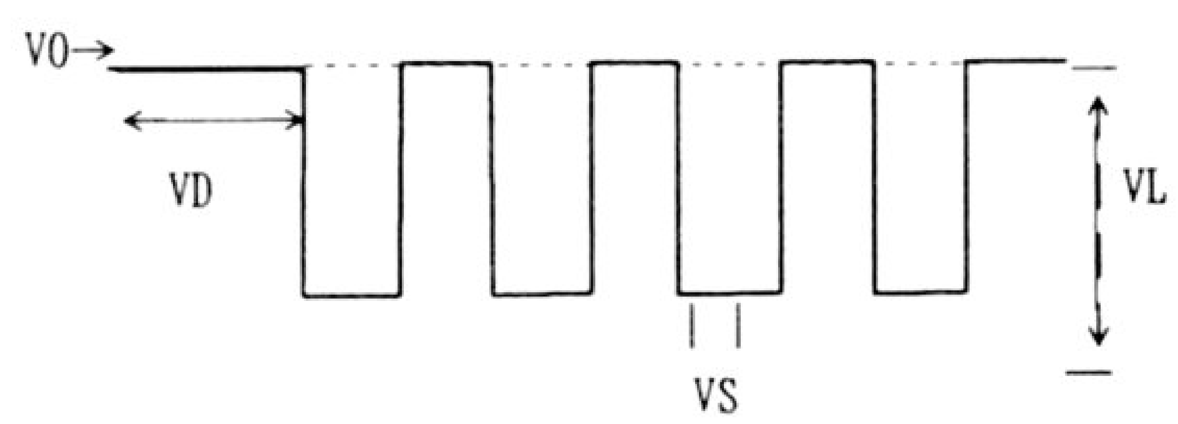
波形番号0:鋸歯波　(振幅が負のとき)



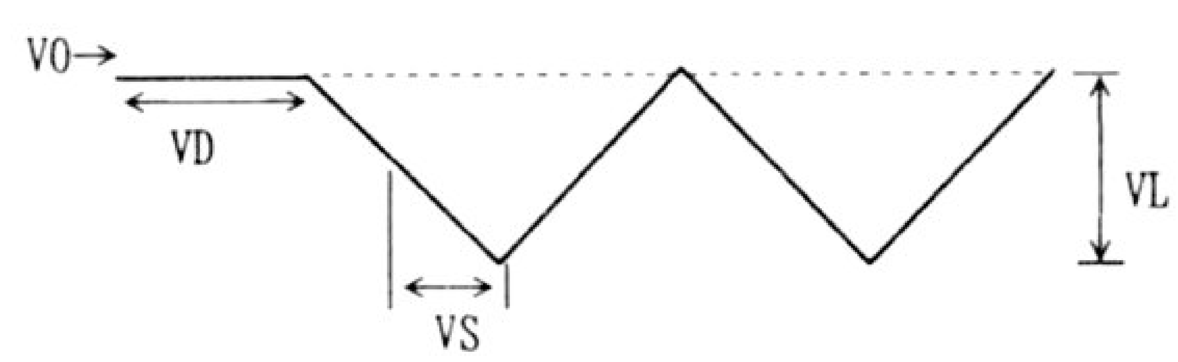
波形番号1:矩形波　(振幅が正のとき)



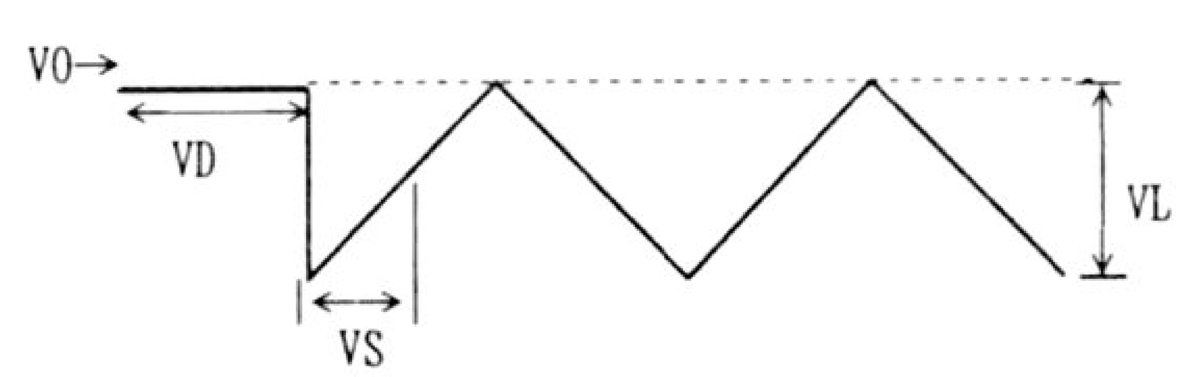
波形番号1:矩形波　(振幅が負のとき)



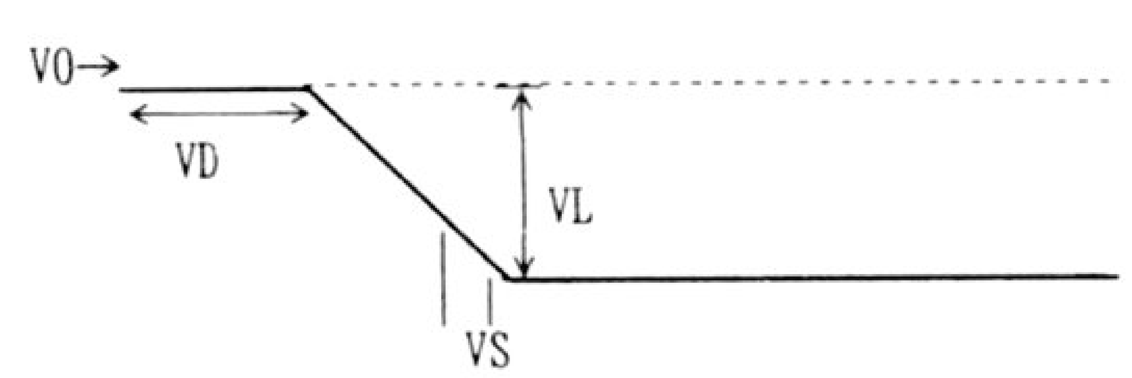
波形番号2:三角波　(振幅が正のとき)



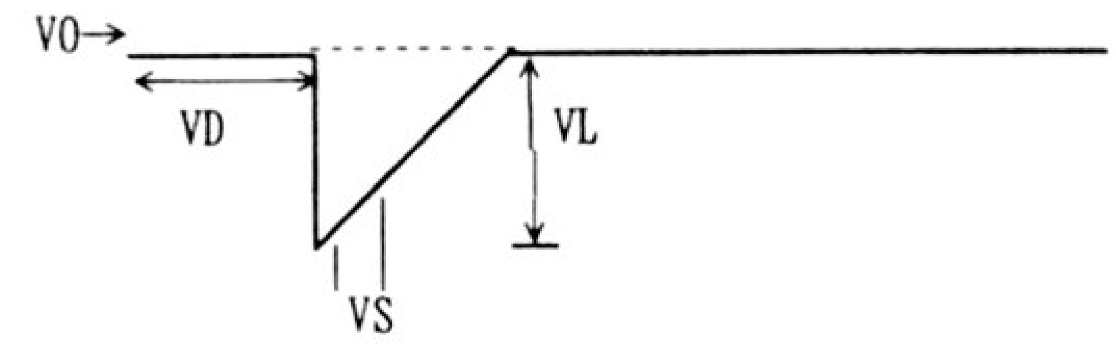
波形番号2:三角波　(振幅が負のとき)



波形番号3:鋸歯波シングル　(振幅が正のとき)

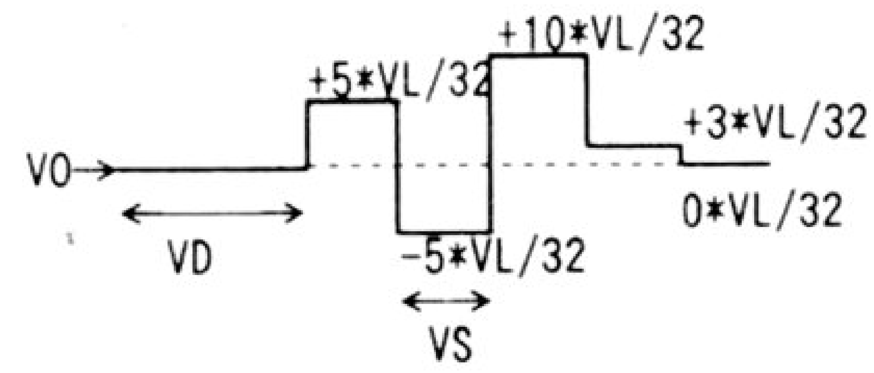


波形番号3:鋸歯波シングル　(振幅が負のとき)



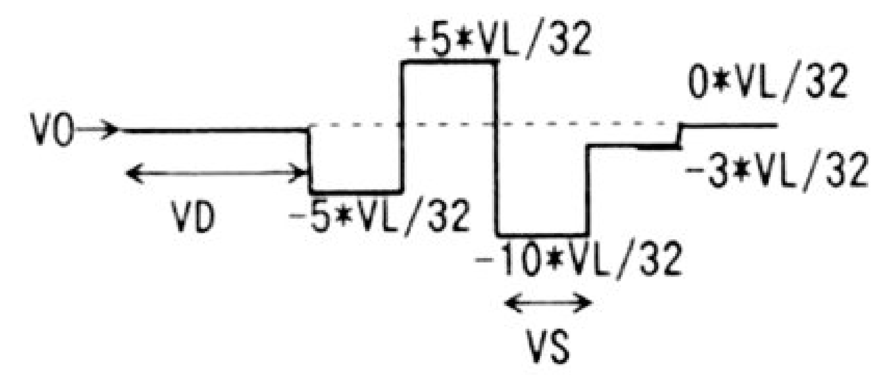
波形番号4:ランダム波　(振幅が正のとき)
波形番号8～32767:ユーザー波形　(振幅が正のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



波形番号4:ランダム波　(振幅が負のとき)
波形番号8～32767:ユーザー波形　(振幅が負のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



FM ADPCM MIDI　EVS波形の原点設定
[VELOCITY.ORIGIN org]

・org:0-127

EVS波形の基準値を設定する。この値を中心に[VELOCITY.DEPTH]([VELOCITY.LEVEL])で設定した振幅で振動する。この値は[VELOCITY.WAVEFORM]の図解中のVOに相当する。初期値は127。

例

[VELOCITY.ORIGIN 100]　　　(EVSの波形の基準値を100に設定する)
　上の設定で(基準点=100)たとえば振幅10のとき
　波形=TRIANGLE(三角波)
　にするとベロシティは100～90(100-10)区間を交互に変化する

FM ADPCM MIDI　EVSスイッチ
[VELOCITY.SWITCH n]

・mode:-1～2　または　OFF,OPTIONAL,ON

EVSの動作制御を行う。

modeは[VELOCITY.DEPTH]([VELOCITY.LEVEL])のパラメータのmodeと同等。modeは数値で与えることもでき、このときは、

0=OFF,1=OPTIONAL,2=ON

に対応する。

例

[VELOCITY.SWITCH OPTIONAL]
(EVSをOPTIONALモードでONにする)

[VELOCITY.SWITCH 2]
(EVSを以前のモードでONにする)

FM ADPCM MIDI　EVS同期モード
[VELOCITY.SYNC m,c]

・m:0=非同期　1=同期　　または　OFF,ON
・c:1～65535

EVS波形の初期化(リセット)タイミングの設定を行う。

mは同期モードの設定値。0は非同期,1は同期に対応する。mをON,OFFといった文字列にて与えることも可能で、この場合、文字列OFF,ONは、それぞれ非同期、同期の設定に対応する。

「同期」に設定した場合は小節線([BAR])のときにEVS波形の初期化を行う。

「非同期」にした場合は発音と無関係にEVSを続行する。

数値指定では0,1以外に、以下のように2進数で設定すればより高度な同期設定が行える。[]は初期設定を表す。

第0ビット　小節線に同期して初期化　　[0]:しない1:する

第1ビット　ベロシティが指定されている音符　[0]:しない1:する
　　　　　があっても波形は進行する
第2ビット　休符があっても波形は進行する　[0]:しない1:する
上からわかるように初期値は0となっている。
cは同期ディレイにあたり、実際にはmで指定した条件がc回成立すると波形が初期化されることになる。cの初期値は1。

例

[VELOCITY.SYNC %011]

(EVSを小節線同期、ベロシティが指定されている音符があっても波形は進行する)

FM ADPCM MIDI　EVS波形位相制御
[VELOCITY.PHASE mode]

・mode:0～1　または NORMAL,REVERSE(INVERSE)

EVS波形の位相を制御する。通常とデフォルト状態ではmode=0(NORMAL)に相当し、[VELOCITY.WAVEFORM]の項で図解した波形が実行される。一方mode=1(REVERSE)では[VELOCITY.WAVEFORM]の項で図解した波形においてV0を中心に反転した波形が実行される。

例

[VELOCITY.PHASE REVERSE]　(EVSの実行波形を逆位相モードにする)

■パンポット

FM ADPCM MIDI　パンポット
[PANPOT p]

・p:0-127 あるいは -127～+127(+,-の符号を明記すると相対指定)

パンポット(定位)を設定する。p=0が最左、p=64が中央、p=127が最右に対応する。pの設定範囲は0≦p≦127。数値の頭に+,-の符号を付記すると相対指定になる。この場合のpの設定範囲は-127≦p≦+127。

またpに文字列、

L63～L1 M R1～R63

を与えることもできる。文字列パラメータと数値パラメータとの対応は以下に示す。

数値	L63～L1 M　R1～R63
文字列	0　～63 64 65～127

例

[PANPOT R20]　(基本MMLの@P84に相当)

FM ADPCM MIDI　相対パンポット
[@PANPOT p]

・p:-127～127

パンポットの設定を相対指定で行う。基本MML @Pに相当する。pの設定範囲は-127≦p≦+127。[PANPOT]とは違い先頭に+,-をつけなくても相対指定として見なされる。

例

[@PANPOT 12]　(それまでのパンポットを12上げる)
[@PANPOT -12]　(それまでのパンポットを12下げる)

■操作チャンネル切り換え

FM ADPCM MIDI　チャンネル切り換え
[CH_ASSIGN ch]
[ASSIGN ch]

・ch:FM1～FM8
　ADPCM1～ADPCM16(またはPCM1～PCM16)
　MIDI1-1～MIDI1-16,MIDI2-1～MIDI2-16,MIDI3-1～MIDI3-16,MIDI4-1～MIDI4-16

そのトラックに割り当てられているデバイス、チャンネルを変更する。Z-MUSIC ver.2.0のMMLであるNや@Nの機能に相当する。

例

[CH_ASSIGN MIDI3-1]　(そのトラックをMIDI3-1に変更する)

■パターン

FM ADPCM MIDI　パターン展開
[PATTERN id]

・id:パターン名文字列

共通コマンド.PATTERN(MEASURE4参照)で登録したパターン名idの演奏データにリンクする。idにはパターン名を記述する。

例

```
(i)
.pattern ABC { cdefgab<c> }
.assign TR1,MIDI1
(t1)[PATTERN ABC]
(p)
```

(cdefgab<c>が演奏される)

FM ADPCM MIDI　パターン展開(埋め込み)
[EMBED id]

・id:パターン名文字列

共通コマンド.PATTERN(MEASURE4参照)で登録したパターン名idの演奏データを展開する。idにはパターン名を記述する。

[PATTERN]は演奏実行時にパターン部にジャンプする形で演奏されるが[EMBED]はパターンの内容そのものをそのトラックに埋め込むので、演奏時は通常の演奏データとまったく変わらない方法で演奏される。

つまり、[PATTERN]は処理速度的には[EMBED]よりも劣るが、演奏データサイズを削減できる。逆に[EMBED]は処理速度的に[PATTERN]より優れるが演奏データサイズは大きくなる。

例

```
(i)
.pattern ABC { cdefgab<c> }
.assign TR1,MIDI1
(t1)[EMBED ABC]
(p)
```

(cdefgab<c>が演奏される)

■テンポ

FM ADPCM MIDI　テンポ
[TEMPO t]

・t:1～32767

テンポを設定する。設定範囲は使用タイマやマスタークロックの値により変動する。

tは1分間の4分音符の数に相当する。

初期値は120。ひとつのトラックで指定すると全トラックに影響する。

例　[TEMPO 200]　(テンポ200を設定)

FM ADPCM MIDI　相対テンポ
[@TEMPO t]

・t:-32768～32767

テンポを相対的に設定する。演算の結果有効範囲を超えてしまった場合は設定可能範囲内に自動修正される。

ひとつのトラックで指定すると全トラックに影響する。

例

[@TEMPO -20]　(現在のテンポに20を引いた値を設定)

FM ADPCM MIDI　タイマ設定
[TIMER t]

・t:0～16383

割り込み周期を割り込みタイマへ直接設定する。基本MMLの@Tに相当する。

以下に各タイマとテンポの相関を表す(マスタークロック=192のとき)。

タイマA＝1024－(78125/テンポ)
タイマB＝256－(78125テンポ)/16
MIDIタイマ=30,000,000/(192*テンポ)

ひとつのトラックで指定すると全トラックに影響する。

例

[TIMER 38]　(タイマ値を38に設定する)

■アゴーギク

FM ADPCM MIDI　アゴーギク振幅
[AGOGIK.DEPTH mode,lv1,lv2,…,lv8]
[AGOGIK.LEVEL mode,lv1,lv2,…,lv8]

・mode:OFF,1/8,OPTIONAL,ON
・lv1～lv8:-32768～+32767

アゴーギクとは演奏速度(テンポ)を一定の法則に従って変動させる効果のことをいう。Z-MUSIC ver.3.0では、ビブラートやARCCのような算術波

形やユーザー波形に沿ってテンポを変化させることができ、これをアゴーギク機能と呼ぶ。

[AGOGIK.DEPTH]([AGOGIK.LEVEL])はアゴーギクの振幅の設定をlv1〜lv8において行う。

modeはアゴーギクの動作方式を設定するパラメータ。以下にmodeと対応する動作を示す。

・mode:ON

アゴーギクを以前の設定で再開する。

・mode:OFF

アゴーギクを停止する。

・mode:1/8

発音する音符の音長の1/8単位ごとでlv1からlv8のアゴーギク振幅を次々に変化していく。

・mode:OPTIONAL

発音する音符の音長とは無関係に、任意の時間区間ごとでlv1からlv8のアゴーギク振幅を次々に設定していく。時間区間のシーケンスは後述の[AGOGIK.DELAY]で行う。

modeは省略するとスイッチはそれまでの設定を保持したままとなる。

lv1〜lv8のアゴーギク振幅は任意の位置で省略可能で、数値の頭に±をつけることによって相対的な値の指定も可能。たとえば、

[AGOGIK.LEVEL optional 64,,-10,+5]

といった記述ができる。省略した部分は前の値を持続するということになり、その間は振幅の変更は行われない。

相対指定時にオーバーフローを起こした値は有効範囲内の最大値,最小値に修正される。

例

[AGOGIK.LEVEL 1/8 0,0,10,20,30]

modeを1/8とし、lv1=0,lv2=0,lv3=10,lv4=20,lv5=30と設定。

lv6,lv7,lv8は省略(未設定)。

FM ADPCM MIDI アゴーギク振幅増加
[AGOGIK.DEEPEN sw,spd,dpn,rpt]

・sw:OFF,ON

・spd:1〜65535

・dpn:-32768〜+32767

・rpt:0〜32767

アゴーギクの振幅の増加/減少を設定する。

・sw:OFF,ON

アゴーギクの振幅増減機能を有効にするか無効にするかを設定する。

OFFで無効に、ONで有効に設定される。

・spd:増減速度

アゴーギクの振幅の増減速度を設定する。1がもっとも速く、65535がもっとも遅い。値は絶対音長に相当し、有効範囲は1〜65535。

・dpn:振幅増加量

振幅の増減量を設定する。負値は振幅減少、正値は振幅増加に対応する。

値の有効範囲は-32768〜+32767。絶対値が大きければ大きいほど、増減は急激となる。

・rpt:増減回数

振幅はspdの速度でdpn幅増減するが、この処理を何回繰り返すかを設定する。

rpt=0とすると振幅が初期化されるまで無限に増減を繰り返す設定になる。値の有効範囲は1〜32767。

例

[AGOGIK.DEEPEN ON,6,+8,10]

(増減スピード6で10回振幅を8増加させる)

FM ADPCM MIDI アゴーギクスピード(周期)
[AGOGIK.SPEED sp1,sp2,…,sp8]

・sp1〜sp8:1〜32767

アゴーギクの振動速度を設定する。最大8つの振幅変化に対応した振動速度変化の設定が行える。設定範囲は1≦sp1〜sp8≦32767。

実際の波形にどう影響するかは拡張MML[AGOGIK.WAVEFORM]の項で解説するが、値が小さいほど高速に、値が大きいほど低速に音が振動する。

sp1〜sp8の振動速度は任意の位置で省略可能だが、初期値を持たないので注意。

例

[AGOGIK.SPEED 10,8,20]

(sp1=10,sp2=8,sp3=20,sp4〜sp8は省略(未設定))

FM ADPCM MIDI アゴーギクディレイ
[AGOGIK.DELAY dl0,dl1,…,dl8]

・dl0:0〜65535

・dl1〜dl8:1〜65535

アゴーギクの動作モードのひとつである「OPTIONAL」モードにおけるアゴーギクの振幅切り換えタイミングを決定する区間時間の設定を行う。

dl0は最初のアゴーギクが実行されるまでのディレイ(遅延時間)に相当し、絶対音長値で指定する。有効範囲は0〜65535。初期値は0。

dl1〜dl8は次の振幅切り換えまでの区間時間で、これも絶対音長値で指定する。有効範囲は1〜65535。初期値はなし。

dl0〜dl8の任意を省略可能だが、dl1〜dl8は初期値を持たないので注意。

例

[AGOGIK.DELAY 10,24,12,12,12,28,28,12,12]

dl0=10,dl1=24,dl2=12,dl3=12,dl4=12,dl5=28,dl6=28,dl7=12,dl8=12が設定される。

FM ADPCM MIDI アゴーギク波形選択
[AGOGIK.WAVEFORM w]

・w:0-32767

アゴーギク波形を選択する。

w=0〜4はプリセット波形に割り当てられている。

・w=0:鋸歯波

・w=1:矩形波

・w=2:三角波

・w=3:ワンショット鋸歯波

・w=4:ランダム波

またはプリセット波形は、

SAW	(w=0:鋸歯波に対応)
SQUARE	(w=1:矩形波に対応)
TRIANGLE	(w=2:三角波に対応)
S.SAW	(w=3:ワンショット鋸歯波に対応)
RANDOM	(w=4:ランダム波に対応)

という文字パラメータを用いて指定することもできる。

初期値はw=2の三角波。

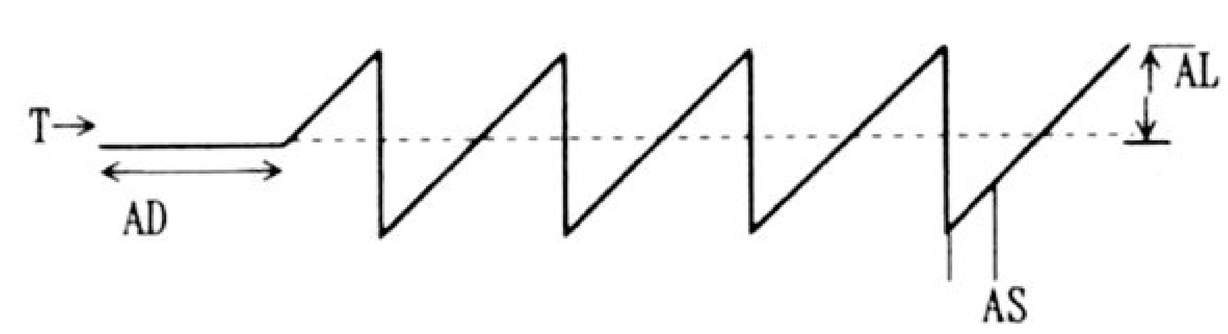
w=5〜7はリザーブ。w=8〜32767はユーザー定義波形となる。

ユーザー波形の登録についてはMEASURE3 zm_set_wave_from1(),MEASURE4「.WAVE_FORM」参照。

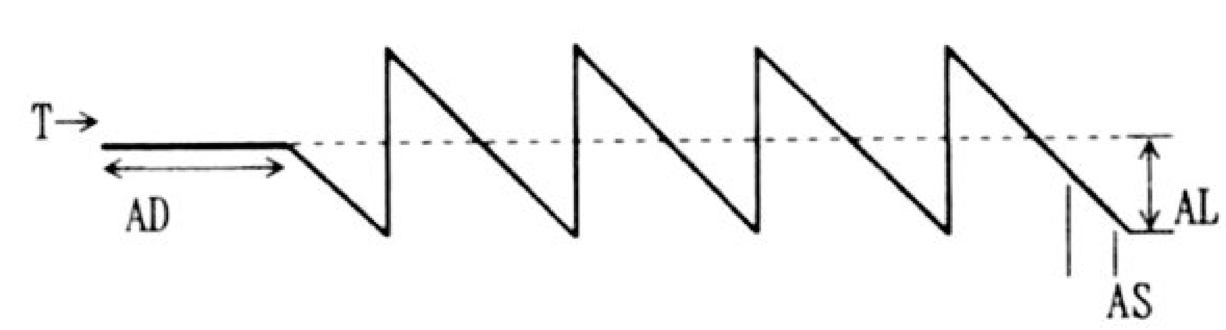
以下に波形と各パラメータとの関係を示す。

AL:振幅	[AGOGIK.DEPTH]([AGOGIK.LEVEL])
AD:ディレイ	[AGOGIK.DELAY]
T:テンポ	[TEMPO]([@TEMPO],T,@T)
AS:速度	[AGOGIK.SPEED]

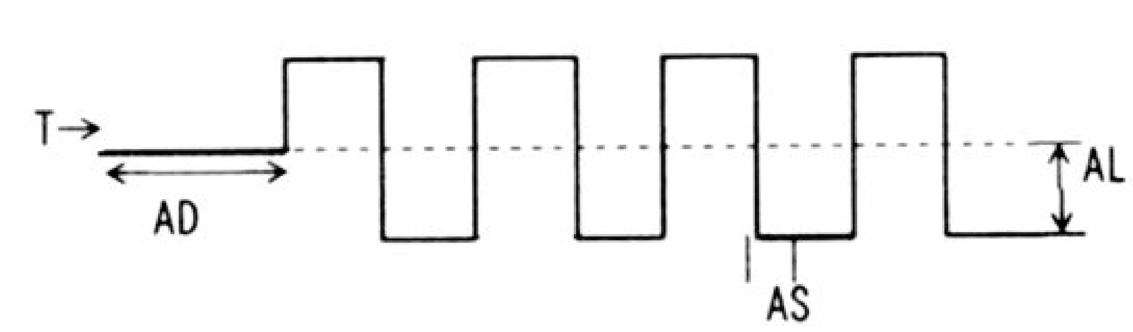
波形番号0:鋸歯波(SAW) (振幅が正のとき)



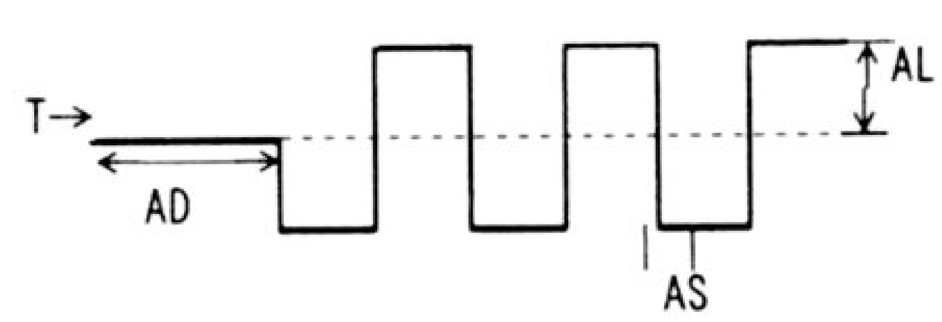
波形番号0:鋸歯波 (振幅が負のとき)



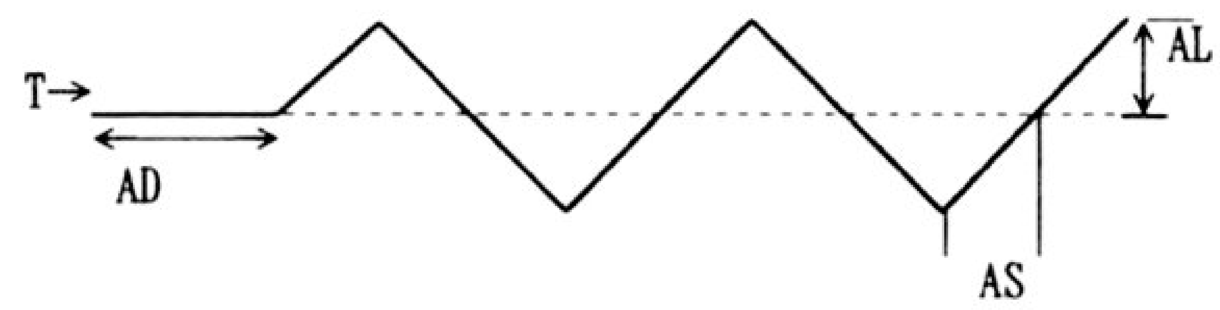
波形番号1:矩形波 (振幅が正のとき)



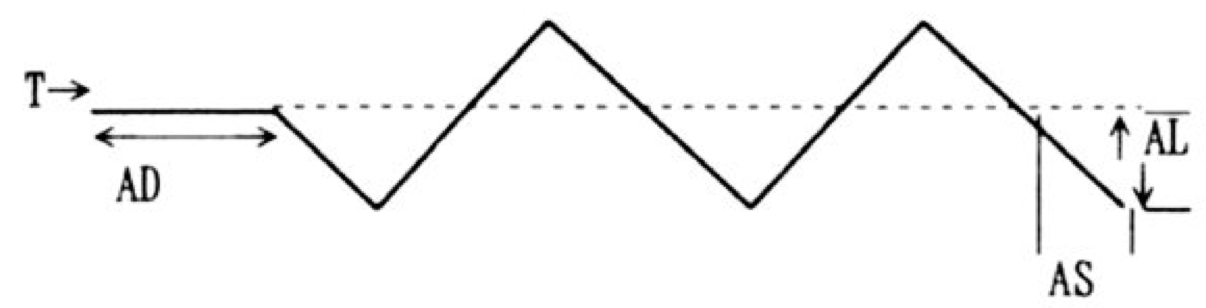
波形番号1:矩形波 (振幅が負のとき)



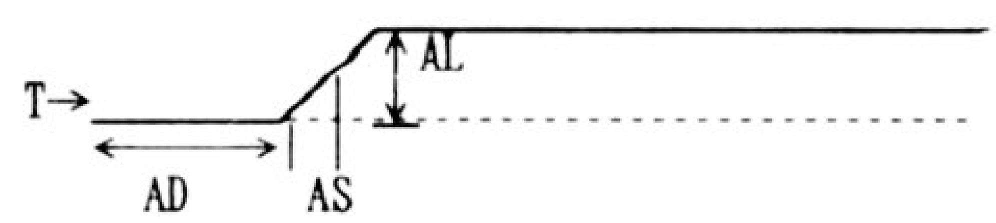
波形番号2:三角波　(振幅が正のとき)



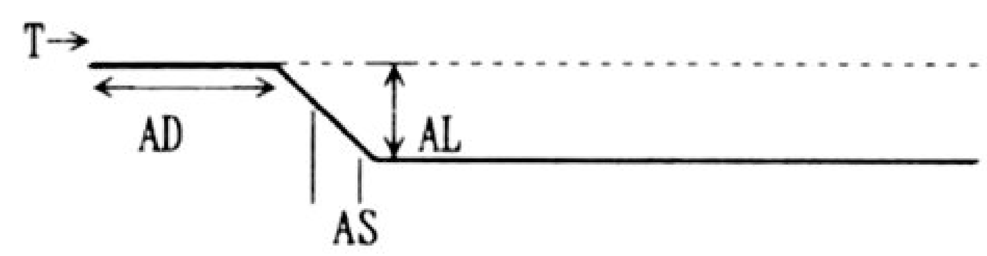
波形番号2:三角波　(振幅が負のとき)



波形番号3:鋸歯波シングル　(振幅が正のとき)

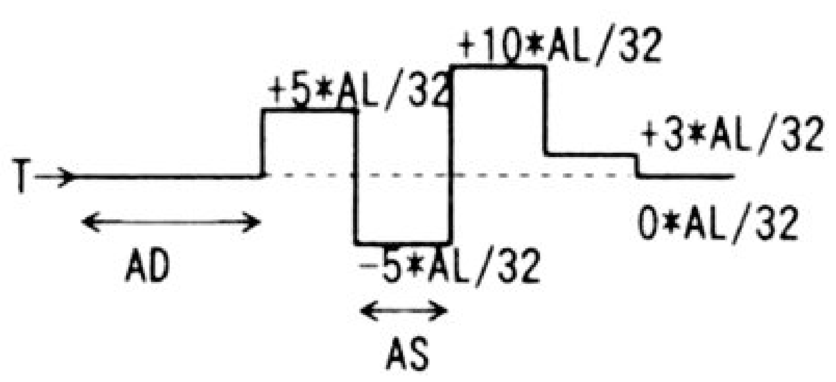


波形番号3:鋸歯波シングル　(振幅が負のとき)



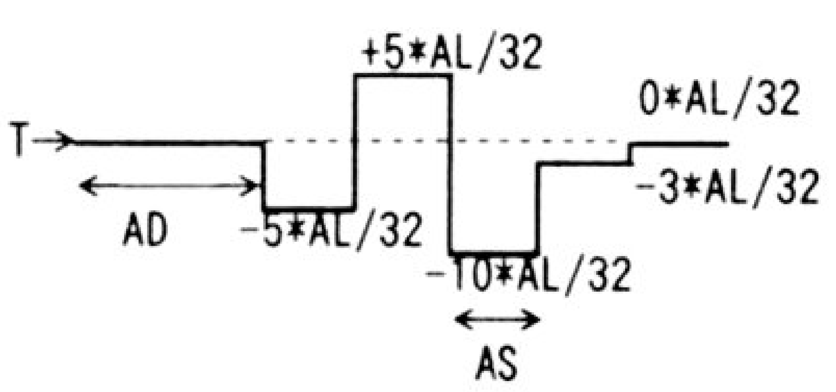
波形番号4:ランダム波　(振幅が正のとき)
波形番号8〜32767:ユーザー波形　(振幅が正のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



波形番号4:ランダム波　(振幅が負のとき)
波形番号8〜32767:ユーザー波形　(振幅が負のとき)

波形データ列が+5,−5,+10,+3,0の場合・・・



例

[AGOGIK.WAVEFORM TRIANGLE]　(AGOGIKの波形を三角波に設定する)

上の設定でテンポ=100で、たとえば振幅が10のときテンポはアゴーギク作用により90〜110間を交互に変化する。

FM ADPCM MIDI　アゴーギクスイッチ
[AGOGIK.SWITCH mode]

・mode:-1〜2　または　OFF,1/8,OPTIONAL,ON

アゴーギクの動作制御を行う。modeは[AGOGIK.DEPTH]([AGOGIK.LEVEL])のパラメータのmodeと同等。modeは数値で与えることもでき、このときは、

0=OFF,-1=1/8,1=OPTIONAL,2=ON

に対応する。

例

[AGOGIK.SWITCH 1/8]
(1/8モードでONにする)

[AGOGIK.SWITCH 2]
(以前のモード(1/8かOPTIONALのいずれか)でONにする)

FM ADPCM MIDI　アゴーギク同期モード
[AGOGIK.SYNC m,c]

・m:0=非同期　1=同期　　または　OFF,ON

・c:1〜65535

アゴーギク波形の初期化(リセット)タイミングの設定を行う。

mは同期モードの設定値。0は非同期,1は同期に対応する。mをON,OFFといった文字列にて与えることも可能で、この場合、文字列OFF,ONは、それぞれ非同期、同期の設定に対応する。

「同期」に設定した場合は新しく発音するたびにアゴーギク波形の初期化を行う。

「非同期」にした場合は発音と無関係にアゴーギクを続行する。

数値指定では0,1以外に、以下のように2進数で設定すればより高度な同期設定が行える。[]は初期設定を表す。

第0ビット	発音と同時にリセット	0:しない[1]:する
第1ビット	タイでもリセット	[0]:しない1:する
第2ビット	休符でもリセット	[0]:NO　1:YES
第3ビット	波形はキーオンごとに進行する	[0]:NO　1:YES
第4ビット	波形は休符でも進行する	[0]:NO　1:YES

上からわかるように初期値は1となっている。

cは同期ディレイにあたり、実際にはmで指定した条件がc回成立すると波形が初期化されることになる。cの初期値は1。

例

[AGOGIK.SYNC %011]　(発音と同時、タイ指定においても初期化する)

■各種モード設定

MIDI　タイモード
[TIE_MODE n]

・n:0〜1　または　NORMAL,SPECIAL(ENHANCED)

MIDIパートのタイの処理モードを設定する。基本MML @Jの機能に相当する。

・n=0　または　NORMAL

通常モードに設定する。

c&d

とした場合、通常モードでは & のない、

c d

として処理され演奏される。

・n=1　または　SPECIAL(ENHANCED)

FM音源パートと互換の動作をするモードに設定される。

c&d

とした場合、FM音源部と同様に、ある音の発音後、その音の持続音(減衰音)で、次の音の音程へ変化するようになる。よって、FM音源部での、

(c<d)&(d>c)

といった表現や、

(g,e)24&e
@b0,683 c4&c+4

のような表記もFM音源部と同様に機能するようになる(ただし、MIDI規格の仕様制限から最初に発音した音高から1オクターブを超えることはできない)。

初期値はn=0(NORMAL)。

例

[TIE_MODE ENHANCED]　(タイモードをFM音源互換モードに設定する)

FM ADPCM MIDI　トラックモード
[TRACK_MODE n]

・n:0,1　または　NORMAL,RHYTHM

そのトラックのノートオフ処理を行うか省略するかどうかの設定を行う。基本MML @Rの機能に相当する。

・n=1　または　RHYTHM

そのトラックは「ノートオフ省略モード」になり、以後ノートオフ処理をしない。

(詳細は基本MML @Rの項を参照)

ノートオフ省略モードでは音符にゲートタイムという概念がなくなるため、MML Qの影響を受けずエンベロープを最後まで実行するようになる。そのため音を細かく区切ったりなどというニュアンスは再現不可能になる。

・n=0　または NORMAL

ノートオフ処理をする通常のモードに設定する。

初期状態は[TRACK_MODE NORMAL]、つまり「ノートオフをする」モードに設定されている。

例

[TRACK_MODE RHYTHM]　(ノートオフ処理を省略する)

FM ADPCM MIDI　トラックディレイ

[TRACK_DELAY n]
・n:0-32768

そのトラックの演奏をnで与える絶対音長分遅らせる。基本MMLの@Wとほぼ同機能の命令。初期値はn=0で、遅れなし。

例

[TRACK_DELAY 10]　（そのトラックを絶対音長10分演奏を遅らせる）

FM ADPCM　トラック同時発音数設定
[VOICE_RESERVE n]
・n:1～8(FM音源パート)
・n:1～16　(AD PCM音源パート)

そのトラックで同時に発音する音数を設定する。n>2に設定した場合は、単音を発音する場合でも別のチャンネルで発音するようになり、MIDI音源のPOLYモードのような効果を得ることができる。

FM音源パートはn=1～8、ADPCM音源パートではn=1～16。初期値はFM音源パート,AD PCM音源パートともにn=1。

例

[VOICE_RESERVE 4]　（最大同時発音数を4に設定する）

FM ADPCM MIDI　強制消音
[ALL_SOUND_OFF]

その時点で発音している音を強制的にすべて消音する。ただし、ダンパーONやキーオフなしモードに設定されている場合はノートオフできない。

例

[ALL_SOUND_OFF]

最大同時発音数について

FM音源パートとAD PCM音源の最大同時発音数設定はMIDI音源のDVA(動的音源確保方式)とは異なり、シーケンシャルにチャンネルをトラックに割り当てる。

たとえば、あるトラックがチャンネル1に割り当てられており、このとき、このトラックの最大同時発音数を4にした場合、チャンネル1,2,3,4が発音に用いられる。

また、FM音源パートにおいて、あるトラックがチャンネル7に割り当てられており、このとき、このトラックの最大同時発音数を4にした場合はチャンネル7,8,1,2が発音に用いられる。

以上のことからチャンネルの競合に注意しないと発音した音が途切れしまう場合がある。

■エコー

FM ADPCM MIDI　エコー
[ECHO itv,dec,lpt]
・itv:音長
・dec:-128～127
・lp time:0～65535

設定後の音符をitv単位に分割して演奏する。分割した音符ひとつひとつに対して線形な元音量との増減decを与えることができる。1チャンネルで疑似的なエコー効果を与えるときなどに使用する。

・itv:音楽的音長

音符を分割する単位を音長(1～32767)で指定する。音長は音楽的音長で指定するが、

数値の頭に'*'を記述すれば絶対音長(1～32767)を指定することもできる。

・dec:減衰相対音量値

分割した音符1つひとつに対して元音量との相対的な差を与えることができる。

分割後n番目の音符の音量の変位はn*decになる。設定範囲は-128～127。

・lp:繰り返し回数

分割を何回繰り返すかを指定する。設定範囲は0～65535。0は無限回を意味する。

また無限回は文字列LOOPでも指定することができる。

例

[ECHO 8,-10,3] c1,,127

この例ではc1は c8 単位に 3回分割される。そして音量変位は-10なので結果的には、

c8,,-10&c8,,-20&c2.,,-30

のように演奏される。

例

[ECHO 8,-10,LOOP] c1

では、分割回数が無限回になっているので c1 をできる限り分割する。よって演奏結果は、

c8,,-10&c8,,-20&c8,,-30&c8,,-40&c8,,-50&c8,,-60&c8,,-70&c8,,-80

のようになる。

エコー機能の仕様制限事項

1.連符コマンド({...}の中)に対しては効力を発揮せず
2.音量操作は具体的にはベロシティを対象としている
3.音符に対してのベロシティが省略されている場合はその音符の直前のベロシティ値が基準となりこの値から減衰する(この場合は直前のベロシティから±63の変化に限定される)。
4.音符にベロシティを絶対値設定している場合はその値を基準として減衰する(0～127の範囲内で増減)。

■調号/拍子

FM ADPCM MIDI　調号
[KEY_SIGNATURE k1,k2,…,kn]
・kn:文字列

調号を設定する。どの音階にどの変化記号（#，♭）がつくのかを設定する。基本MML [K.SIGN]とまったく同等の機能。

kn部分は変化記号、音階の順に設定し、1個ずつ　，で区切る。

例

[K.SIGN +c,+d,+f,+g]　(ホ長調)

cdfgに自動的に#(シャープ)がつく

[K.SIGN -a,-b,-d,-e]　(変イ長調)

abdeに自動的に♭(フラット)がつく

[K.SIGN -a,+b]

aに♭(フラット)が、bに#(シャープ)が自動的につく

また以下の調名を記述することによって、一般的な調号の設定を行うこともできる。

Cmajor,Gmajor,Dmajor,Amajor,Emajor,Bmajor,F+major,F#major,C+major,C#major,Fmajor,B-major,Bbmajor,E-major,Ebmajor,A-major,Abmajor,D-major,Dbmajor,G-major,Gbmajor,C-major,Cbmajor,Aminor,Eminor,Bminor,F+minor,F#minor,C+minor,C#minor,G+minor,G#minor,D+minor,D#minor,A+minor,A#minor,Dminor,Gminor,Cminor,Fminor,B-minor,Bbminor,E-minor,Ebminor,A-minor,Abminor

例

[K.SIGN G-Major]　変ト長調(cdegabに♭)

ナチュラル記号である'!'を使用すれば一時的に調号の効力をなくすことができる。

また、調号の効力をなす音階に対してさらに#,+(#)や-(♭)がついた場合は調号と両方考慮した音階になる。

例

[K.SIGN G-Major] b!

(この調ではbには通常ならば♭がつき、b-になる(半音下がる)のだが！のためナチュラル指定となりbで演奏される)

[K.SIGN G-Major] b-

(この調ではbには通常ならば♭がつき、b-になる(半音下がる)のだが♭のためb--(=a)で 演奏される)

FM ADPCM MIDI
[KEY　ky]
・ky:文字列

曲全体の調を設定する。ただし演奏トラックに調号のような効果や機能はなく、あくまで表向きの設定となる。

kyには以下の調名を与える。

Cmajor,Gmajor,Dmajor,Amajor,Emajor,Bmajor,F+major,F#major,C+major,C#major,Fmajor,B-major,Bbmajor,E-major,Ebmajor,A-major,Abmajor,D-major,Dbmajor,G-major,Gbmajor,C-major,Cbmajor,Aminor,Eminor,Bminor,F+minor,F#minor,C+minor,C#minor,G+minor,G#minor,D+minor,D#minor,A+minor,A#minor,

Dminor,Gminor,Cminor,Fminor,B-minor,Bbminor,E-minor,
Ebminor,A-minor,Abminor

例

```
[KEY D#Minor]                    (嬰二短調)
```

FM ADPCM MIDI　拍子の設定

[METER m]

・1/1～255/255

曲全体の拍子を設定する。ただし演奏トラックに対してなんの効果ももたらさない。あくまで表向きの設定となる。設定値mは分数表記を取る。

例

```
[METER 6/8] (8分の6拍子と設定)
```

■小節

FM ADPCM MIDI　小節線

[---------------] ('-'はいくつでもよい)

[BAR str]

[MEASURE str]

・str:コメント文字列

五線譜でいう小節線に相当する。命令自体に機能はない。strはZMSを見やすくするためにコメント文字列を与えることができる(ただしstrの情報はZMDには残らない)。

トラックの初期状態は第０小節に相当する。

例

```
[MEASURE Aパート]
[----------------]
```

FM ADPCM MIDI　小節ジャンプ

[JUMP m]

・m:1～

mで表される小節の先頭へジャンプする。ジャンプ後の音源やZ-MUSICの内部パラメータはそこまで通常に演奏を行ったときと同じになっている。基本MMLの[@]とほぼ同等の機能を持つ。該当小節がない場合は正常に演奏されない。

例

```
(i)
.assign TR1,MIDI1
(t1)@1 v12
(t1)[jump2]     /第2小節から演奏
(t1)cde         /第0小節
(t1)[bar]
(t1)fga         /第1小節
(t1)[bar]
(t1)b<cd        /第2小節
(t1)[bar]
(p)
```

(この例では b<cd のみが演奏される。)

■演奏制御

FM ADPCM MIDI　再演奏

[REPLAY t]

・t:1-65535

トラックtを強制的に初めから演奏開始にする。基本MML Jに相当する。演奏が終了していても、演奏中であっても演奏開始を受けたトラックは最初から演奏を始める。設定範囲は1≦t≦65535。

例

```
(t1)  @1 v15 o4 q8 [REPLAY 2]r2[REPLAY2]
(t2)  @1 v15 o4 q8 l8cdef
```

この例では結果的にはcdef cdefが演奏される。

FM ADPCM MIDI　同期

[SYNCHRONIZE t]

・t:1-65535

トラック間同期通信を行う。基本MML Wに相当する機能を持つ。tはトラック番号を表し、この値の有効範囲は1～65535。トラックtに対して同期信号を送信する。同期信号を送られたトラックは直ちに演奏を再開する。

値tを省略すると、これは「同期待ち」の指示となり、直ちに演奏を中断し、ほかのトラックから同期信号が送られるまで演奏処理を停止続ける。

なお、本命令は設計の都合上、[@],[!],[JUMP]といったジャンプ系コマンドと併用できない点に注意。

例

```
(t1)                 @1 v15 o4 q8 cdef [synchronize 2]
                                        ↑トラック２へ同期送信
(t2) [synchronize]   @1 v15 o4 q8 gab<c
         ↑どこからか同期信号がくるまで演奏を中断。
```

この例では結果的にはcdefgab<cが演奏される。

■ワーク書き換え

FM ADPCM MIDI　ワーク書き換え

[POKE a,d]

・a:-2147483648～2147483647

・d:0～4294967296 または -2147483648～2147483647。

そのトラックの演奏ワークのａ番に値ｄを書き込む。Z-MUSICの演奏トラックワークａに値ｄを設定する。基本MMLの？に相当する。ｄは数値の頭に±をつけて相対値の設定も可能。

使用にはZ-MUSICのワークエリアについての十分な理解を必要とする。使用方法によっては最悪、暴走してしまうので注意。

例

```
[POKE -10,+50]
```

(演奏トラックワークの先頭-10の内容に+50加算する)

■ダミー

FM ADPCM MIDI　ダミー

[DUMMY]

なにも行わない命令。ZMSコンパイル後のZMDに1バイトの空きスペースを作ることができる。通常の音楽制作には用いる必要はない。

■コメント

FM ADPCM MIDI　コメント

[COMMENT str]

・str:文字列

コメントを与える。コンパイル後のZMDにもコメント文字列が内部的に埋め込まれる点が基本MML /と違う。

例

```
[COMMENT This is a pen.]
```

■イベント

FM ADPCM MIDI　イベント

[EVENT category,class,data]

・category:WORD,PICTURE,SOUND

・class:文字列

・data:文字列

演奏データに外部プレイヤー向けのイベント情報を盛り込むときに使用する。Z-MUSIC ver.3.0自身は本命令の実行は行わず無視する。

・category:イベント種類

イベントの種類を指定する。現在定められているのは、

WORD:歌詞などの文字列データ
PICTURE:グラフィックデータ
SOUND:効果音など

の３つ。

・class:具体的なデータフォーマットタイプ

categoryで指定したイベントの種類を具体的にどのようなデータ形式を用いて表現するかを文字列で指定する。現時点では各categoryにつき１種類のclass のみ定義されている。以下にそれらを示す。

category=WORDのとき	class=STRING	歌詞文字列
category=PICTUREのとき	class=PIC PIC画像ファイルのファイル名文字列	
category=SOUNDのとき	class=ADPCM AD PCMファイルのファイル名文字列	

例

```
[EVENT WORD,STRING,"豚もおだてりゃ木に登る"]     (歌詞設定)
[EVENT PICTURE,PIC,FISH.PIC]                    (画像データ設定)
[EVENT SOUND,ADPCM,BOMB.PCM]                    (AD PCMデータ設定)
```

■アフタータッチメッセージ

FM ADPCM MIDI　　チャンネルプレッシャー
[CH_PRESSURE n]

・n:0～127　または　-127～+127　(+,-の符号を明記すると相対指定になる)

チャンネルプレッシャーとはチャンネル単位で設定するアフタータッチ機能をいう。本命令はMIDIパートにおいてこのチャンネルプレッシャーのMIDIメッセージを送信するもの(チャンネルプレッシャーの設定値に対応する効果はMID音源側であらかじめ設定しておく必要がある)。

設定範囲は0～127,数値の頭に±を書いて相対指定を行うこともできる。この場合の設定範囲は-127～+127。

なお、FM音源パート、ADPCM音源パートではベロシティの設定(拡張MML[VELOCITY])と同等の機能を果たす。

例

[CH_PRESSURE 64]　　(チャンネルプレッシャーを64に設定する)

FM ADPCM MIDI　　ポリフォニックプレッシャー
[POLYPHONIC_PRESSURE ky,vl]

・ky:0-127 または 0-1C～09G,C-1～G9

・vl:0-127 または -127～+127　(+,-の符号を明記すると相対指定)

ポリフォニックプレッシャーとはノート別に設定することができるアフタータッチ機能のことをいう。本命令はMIDIパートにおいて、このポリフォニックプレッシャーのMIDIメッセージを送信するもの(ポリフォニックプレッシャーの設定値に対応する効果はMID音源側であらかじめ設定しておく必要がある)。

ポリフォニックプレッシャーを与えるノートはkyにて指定する。kyの設定範囲は0～127。kyは0-1C～09G,C-1～G9といったMMLで与えることもできる。

ポリフォニックプレッシャー値であるvlの設定範囲は0～127。数値の頭に±を書いて相対指定を行うこともできる。この場合の設定範囲は-127～+127。

なお、FM音源,音源パートではその時点で発音している指定ノートkyに対して音量(ベロシティ)vlを設定する動作をする。

例

[POLYPHONIC_PRESSURE o4c,64]

(ノート04Cにポリフォニックプレッシャー値64を設定する)

■コントロールチェンジ/NRPN

MIDI (FM ADPCM)　　コントロールチェンジ設定
[CONTROL c,d]

・c:0-127

・d:0-127

MIDI楽器のコントロールチェンジcにデータdを送信する。基本MMLのYに相当する。FM音源パート,AD PCM音源パートではFM音源レジスタcにデータdを設定する。

例

[CONTROL 11,127]

(コントロールチェンジ11番(EXPRESSION)に127を設定する)

MIDI　　NRPN設定
[NRPN MSB,LSB,d1,d2]

・MSB:0-127

・LSB:0-127

・d1:0-127

・d2:0-127

MSB,LSBで表されるMIDI楽器のNRPNアドレスにデータd1,d2を送信する。基本MMLの@Yに相当する。MSB,LSBはそれぞれNRPNのアドレス上位下位を表す。

d1,d2はデータの上位下位を表す。FM音源パート,AD PCM音源パートでは無視される。

d1,d2のみ省略が可能。

例

[NRPN 1,8, $40]

(NRPN $0108にデータ$40**を設定。下位バイトは省略)

■ホールド(ダンパーペダル)

FM ADPCM MIDI　　ホールド(ダンパーペダル)
[DAMPER d]

・d:0-127

ダンパーペダルの設定を行う。基本MMLの@Dに相当する。設定範囲は0-127。一般のMIDI楽器では0-63をダンパーOFF,64-127をダンパーONとしている。ダンパーON,OFFは文字列ON,OFFでも設定が可能。このときはONがd=127,OFFがd=0に対応する。

例

[DAMPER OFF]　　(ダンパー解除する)

[DAMPER 68]　　(ダンパーを値68でONに設定する)

■フェーダー

FM ADPCM MIDI　　トラックフェーダー
[TRACK_FADER spd,src,dst]

・spd:0-65535

・src:0-128

・dst:0-128

トラックフェーダーの動きを設定する。基本MML ¥とほぼ同等の機能を持つ。

・spd:移動速度

フェーダーの移動速度を指定する。spd=1がもっとも遅く、spd=65535がもっとも早い。spd=0は移動解除に相当する。また、移動解除指定はOFFという文字パラメータでも指定可能。

・src:移動元フェーダーレベル

移動元のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

・src:移動先フェーダーレベル

移動先のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

例

[TRACK_FADER 10,128,0]

(速度10で原音量から無音状態へフェーダーを移動する(いわゆるFADEOUT))

FM ADPCM MIDI　　チャンネルフェーダー
[CH_FADER dev ch,spd,src,dst]

・dev:ALL,FM,ADPCM,MIDI1-,MIDI2-,MIDI3-,MIDI4-

・ch:1-16　または　ALL

・spd:0-65535

・src:0-128

・dst:0-128

指定したチャンネルのフェーダーの動きを設定する。

・dev:デバイス

操作の対象となる音源デバイス名を文字列にて指定する。デバイス名と対応音源は以下の通り。

パラメータ	操作対象音源
ALL	全デバイス
FM	FM音源
ADPCM(PCM)	AD PCM音源
MIDI1-	MIDI出力ポート1
MIDI2-	MIDI出力ポート2
MIDI3-	MIDI出力ポート3
MIDI4-	MIDI出力ポート4

・ch:操作対象チャンネル番号

上記のデバイスと合わせて指定する。FM音源パートのチャンネル設定範囲は1≦ch≦8。

AD PCM音源パート、MIDI音源パートのチャンネル設定範囲は1≦ch≦16。

全チャンネルを対象としたい場合は文字列 ALL を指定する。

なお、全デバイスの全チャンネルを操作対象とする場合は ALL-ALLではなく、ALLと指定する。

・spd:移動速度

フェーダーの移動速度を指定する。spd=1がもっとも遅く、spd=65535がもっとも早い。

spd=0は移動解除に相当する。また、移動解除指定はOFFという文字パラメータでも指定可能。

・src:移動元フェーダーレベル

移動元のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

・src:移動先フェーダーレベル
移動先のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

例

[CH_FADER MIDI1-10,10,128,0]
(MIDI出力ポート1の10チャンネルを速度10で原音量から無音状態へフェーダーを移動する(いわゆるフェードアウト))

[CH_FADER FM-ALL,0,64,64]
(FM音源全チャンネルをフェーダーレベル64に設定し固定する)

[CH_FADER ALL-2,0,64,64]
(全デバイスのチャンネル2をフェーダーレベル64に設定し固定する)

[CH_FADER ALL,10,0,128]
(全デバイスの全チャンネルを無音状態から原音量までフェーダーを移動させる。(いわゆるフェードイン))

FM ADPCM MIDI　マスターフェーダー

[MASTER_FADER dev,spd,src,dst,...]

・dev:ALL,FM,ADPCM,MIDI1,MIDI2,MIDI3,MIDI4
・spd:0〜65535
・src:0-128
・dst:0-128

マスターフェーダーの動きを設定する。

・dev:デバイス
操作の対象となる音源デバイス名を文字列にて指定する。デバイス名と対応音源は以下の通り。

パラメータ	操作対象音源
ALL	全デバイス
FM	FM音源
ADPCM(PCM)	AD PCM音源
MIDI1	MIDI出力ポート1
MIDI2	MIDI出力ポート2
MIDI3	MIDI出力ポート3
MIDI4	MIDI出力ポート4

・spd:移動速度
フェーダーの移動速度を指定する。spd=1がもっとも遅く、spd=65535がもっとも早い。
spd=0は移動解除に相当する。また、移動解除指定はOFFという文字パラメータでも指定可能。

・src:移動元フェーダーレベル
移動元のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

・src:移動先フェーダーレベル
移動先のフェーダーレベルを設定する。設定範囲は0≦src≦128で、0が無音状態、128は原音量に相当する。

例

[MASTER_FADER MIDI1,10,128,0]
(MIDI出力ポート1を速度10で原音量から無音状態へフェーダーを移動する(いわゆるフェードアウト))

[MASTER_FADER FM,0,64,64]
(FM音源をフェーダーレベル64に設定し固定する)

[MASTER_FADER ALL,10,0,128]
(全デバイスを無音状態から原音量までフェーダーを移動させる。(いわゆるフェードイン))

> ### フェーダーについて
>
> Z-MUSIC ver.3.0ではトラックフェーダー、チャンネルフェーダー、マスターフェーダーの3つのフェーダーがあります。このうちトラックフェーダーとチャンネルフェーダーは、操作対象とするフェーダーの指定方法が違うだけで、処理方式も機能も同一です。つまり、たとえばトラックフェーダーが実行されているトラックのチャンネルに対してフェーダー制御を行うとそれまで実行されていたトラックフェーダーは無効となりチャンネルフェーダーとして新たな設定で制御されるようになります。
>
> 一方マスターフェーダーはまったく前者2つのフェーダーとは独立した処理系で実行されます。よってトラックフェーダー/チャンネルフェーダーと同時に独立した制御を行うことができます。
>
> このことからトラックフェーダー／チャンネルフェーダーは、トラックごとの演奏表現の一手段として使用し、マスターフェーダーは曲全体の音量制御に利用するのが効果的な使用法といえます。
>
> (マスターフェーダーは外部アプリケーションから曲側の都合を無視して制御されることがあります。)

■FM音源専用命令

FM　FM音源レジスタ書き込み

[OPM name,d]

・name:レジスタ名文字列
・d:0-255

nameで表されるFM音源レジスタに値dを書き込む。以下にレジスタ名一覧を示す。

グローバルレジスタ名	NFRQ,NE,LFRQ,PMD,AMD,WF
チャンネルレジスタ名	CON,FB,PAN,AMS,PMS
オペレータレジスタ名	MUL,DT1,TL,AR,KS,1DR,AMS,2DR,DT2,RR,1DL

なお、オペレータ名は　OP1.xxx〜OP4.xxxのように指定する。
[ADPCM][MIDI]パートで指定された場合は無視される。

例

[OPM LFRQ,150]　　(FM音源レジスタFRQ($18)に値150を書き込む)
[OPM OP2.TL,10]　　(FM音源レジスタOP2のTLに値10を書き込む)

FM　FM音源ハードウェアLFO設定

[OPM.LFO wf,sync,spd,pmd,amd,pms,ams]

・wf:0〜3　　または　SAW,SQUARE,TRIANGLE,RANDOM
・sync:0,1　　または　OFF,ON
・spd:0-255
・pmd:0-127
・amd:0-127
・pms:0-7
・ams:0-3

FM音源のハードウェアLFOのパラメータの設定を行う。Z-MUSIC ver.3.0が提供するソフトウェアLFOとは独立してかけることができる。
設定は基本MMLの音色切り換えコマンドである@で音色パラメータの値に更新されてしまう。(ハードウェアLFOの設定を保存して音色切り替えを実行したい場合は[TIMBRE]命令を用いるとよい)

・wf:波形番号
波形タイプを設定する。設定範囲は0〜3。またはSAW,SQUARE,TRIANGLE,RANDOMの文字パラメータを設定することもできる。以下に数値と文字パラメータの対応を示す。

wf	文字パラメータ	波形名
0	SAW	鋸歯波
1	SQUARE	矩形波
2	TRIANGLE	三角波
3	RANDOM	ランダム波

・sync:LFO初期化設定
キーオンと同期してLFOを初期化するかどうかを設定する。0で初期化しない、1で初期化する。またOFF,ONの文字パラメータで設定することもできる。OFFが初期化しない、ONが初期化する、に対応する。

・spd:LFO速度
ハードウェアLFOの速度の設定を行う。値の有効範囲は0〜255。

・pmd:ピッチモジュレーション振幅
ハードウェアLFOのピッチモジュレーションの振幅の設定を行う。
値の有効範囲は0〜127。

・amd:アンプリチュードモジュレーション振幅
ハードウェアLFOのアンプリチュードモジュレーションの振幅の設定を行う。
値の有効範囲は0〜127。

・pms:ピッチモジュレーション感度
ハードウェアLFOのピッチモジュレーションの感度の設定を行う。
値の有効範囲は0〜7。

・ams:アンプリチュードモジュレーション振幅感度
ハードウェアLFOのアンプリチュードモジュレーションの感度の設定を行う。
値の有効範囲は0〜7。

パラメータsync,pms,amsはチャンネルごとに独立した設定が行えるが、これ以外パラメータ(spd,pmd,amd)の設定は、FM音源の仕様制限から、全チャンネルに影響してしまう。

例

　[OPM.LFO SQUARE,ON,120,88,50,5,2]
　波形タイプを矩形波、速度を120,pmd=88,amd=50,pms=5,ams=2として設定する。

FM　FM音源スロットモード設定
[SLOT_SEPARRATION s]

・s=%0000〜%1111

　FM音源は1チャンネルにつき4つのオペレータを持っており、この４つのオペレータを組み合わせて音を作り出している。本命令はそのトラックに割り当てられているチャンネルの4つのオペレータのうち、どれを発音に用いるかを設定する。この機能をZ-MUSIC ver.3.0ではスロットセパレーション(SLOT_SEPARRATION)と呼んでいる。

　応用次第によってはFM音源1チャンネルで各オペレータ別々に発音させることもできる。

　パラメータsの設定範囲は%0000〜%1111。第0ビットがオペレータ1、第1ビットがオペレータ2、第2ビットがオペレータ3、第3ビットがオペレータ4に対応する。ビット=1のオペレータをそのトラックで使用する。s=%0000あるいは、文字列でOFFを指定した場合は、スロットセパレーション機能を解除する設定となる。

例
　[SLOT_SEPARRATION %1010]　　(発音にOP2,OP4を使用する)
　[SLOT_SEPARRATION OFF]　　(スロットセパレーション機能を解除)

例

```
(i)
/       AR DR SR RR SL  OL KS ML D1 D2 AM        SQU
(@1     31  0  0 15  0  25  0  2  0  0  0
        31 29  0 15  0  35  0  3  0  0  0
        31 29  0 15  0  35  0  5  0  0  0
        31 29  0 15  0   0  0  1  0  0  0
/       AL FB
         5  7)
.assign tr1 {fm1,128,1}
.assign tr2 {fm1,128,1}

(t1)@1v16 [slot_separation %1001] cde
(t2)@1v16 [slot_separation %0110] r8cde
(p)
```

　使用しているチャンネルはFM音源のチャンネル1だけだが、あたかも2チャンネル使用しているような感じが表現できる。

FM　FM音源ノイズ設定
[NOISE n]

・n:0-31　または　OFF

　FM音源のチャンネル８のオペレータ４(スロット32)からノイズを発生する。

　nとノイズの周波数とは、

　　　n=4000/(32×ノイズ周波数[kHz])

の関係があり、設定範囲は0≦n≦31。OFFを指定した場合はノイズモード解除と見なす。

■AD PCM(PCM)関連

ADPCM　PCMモード設定
[PCM_MODE n]

・n:0〜1　または　RHYTHM,TONE,TIMBRE

　Z-MUSIC ver.3.0ではMPCM.X使用時、AD PCM(PCM)音を楽器音(TIMBRE)として登録するこ」とで、FM音源,MIDIパートのように旋律を演奏することができるようになった。このモードをZ-MUSIC ver.3.0ではTIMBREモードと呼ぶ。

　本命令はADPCMパートを従来通り各ノートに割り振りって使用するか(これをリズムモードあるいはトーンモードと呼ぶ)、楽器音として使用するかを選択設定するもの。

・n=0　または　RHTHM,TONE
　AD PCM音源をリズムモード(トーンモード)で使用する。
・n=1　または　TIMBRE
　AD PCM音源をティンバーモードで使用する。
　初期設定はn=0(RHYTHM,TONE)のリズムモード(トーンモード)。

例
　[PCM_MODE TIMBRE]　(トラックをTIMBREモードに設定)

ADPCM　周波数切り換え
[FREQUENCY n]

・n:0〜6

　内蔵AD PCM音源の再生周波数を変更する。基本MMLの@Fの機能に相当する。設定範囲は0≦n≦6で、各値は、

```
n=    0: 3.9kHz
      1: 5.2kHz
      2: 7.8kHz
      3:10.4kHz
      4:15.6kHz
      5:20.8kHz     *1
      6:31.2kHz     *1
```

に対応する。初期値は4で値省略時は4に自動設定される。
*1の周波数はAD PCMチップ供給クロックに対して改造を行ったX680x0に対してのみ有効。

■エフェクト

MIDI　エフェクト
[EFFECT rvb,cho,dly]

・rvb:0-127
・cho:0-127
・dly:0-127
・ef2:0-127
・ef5:0-127

　MIDI楽器のエフェクトを制御する。基本MML @Eに相当する機能。コントロールチェンジのエフェクト１(GSではREVERB)へrvbを、エフェクト３(GSではCHORUS)へchoを、エフェクト４(GSではDELAY)へdlyを出力する。

　このほか、MIDI規格で定められているエフェクト２，エフェクト５にも値が設定可能(よく使われるエフェクト1,3,4を先頭に先頭に持ってきてある)。ただし、楽器側がこれらのエフェクト制御に対応していない場合は、無意味。

　パラメータは任意のものを省略できる。パラメータの有効範囲は0〜127。初期値は不定。

　FM音源パートやAD PCMパートで使用しても無視される。

例
　[EFFECT 0,10,5]　　(REVERBを0,CHORUSを10,DELAYを5に設定する)

MIDI　リバーブ
[EFFECT.REVERB rvb]

・rvb:0-127

　リバーブ(エフェクト1)のかかり具合を設定する。値の設定範囲は0〜127。MIDI楽器側がこれらのエフェクト制御に対応していない場合は、無意味。また、FM音源パートやAD PCMパートで使用しても無視される。

例
　[EFFECT.REVERB 127]（リバーブの設定を127にする)

MIDI　コーラス
[EFFECT.CHORUS cho]

・cho:0-127

　コーラス(エフェクト3)のかかり具合を設定する。値の設定範囲は0〜127。MIDI楽器側がこれらのエフェクト制御に対応していない場合は、無意味。また、FM音源パートやAD PCMパートで使用しても無視される。

例
　EFFECT.CHORUS 127]　(コーラスの設定を127にする)

MIDI　ディレイ
[EFFECT.DELAY dly]

・dly:0-127

　ディレイ(エフェクト4)のかかり具合を設定する。値の設定範囲は0〜127。MIDI楽器側がこれらのエフェクト制御に対応していない場合は、無意味。また、FM音源パートやAD PCMパートで使用しても無視される。

例
　[EFFECT.DELAY 127]　(ディレイの設定を127にする)

■MIDIデータ送信

MIDI　MIDIデータ送信
[MIDI_DATA n1,n2,…]

・ni:0-127

MIDIデータの送信を行う。基本MML @Xに相当する機能。メーカーや楽器の種類には無関係でチェックサムなども発信しない。

データはいくつでも記述可能。各値の設定範囲は0≦xi≦255であるが、256以上の値を設定した場合は7ビットずつに分割されて送信される(8ビットでない点に注意)。

(一般的なMIDI規格の慣例に従い、下位、上位の順に送信される。例:8192→$00,40)

また、"～"で括ればその文字コードを送信できる。ただし、同機能の共通コマンド.MIDI_DATAとは異なり複数行にパラメータを分けて書くことはできない。

本命令は楽器のメモリを書き換える場合もあるので各自の責任の下で使用すること。

例

[MIDI_DATA $b0,11,127]	($b0,11,127を送信する)
[MIDI_DATA "ABC"]	($41,$42,$43を送信する)

MIDI　ローランドエクスクルーシブ
[ROLAND_EXCLUSIVE n1,n2,…]

・ni:0-127

ローランド系の楽器にエクスクルーシブメッセージ送信を行う。基本MML Xと同等の機能。

後述の[INSTRUMENT_ID]でメーカーIDをローランド($41)に設定しておくことが前提。 データの転送方式は「ワンウェイコミュニケーションのデータセット」(コマンドID=＄12)。

データはいくつでも記述可能。各値の設定範囲は0≦ni≦127であるが、128以上の値を設定した場合は7ビットずつに分割されて送信される(一般的なMIDI規格の慣例に従い、下位、上位の順に送信される。例:8192→$00,40)。

また、"～"で括ればその文字コードを送信できる。ただし、同機能の共通コマンド.ROLAND_EXCLUSIVEとは異なり複数行にパラメータを分けて書くことはできない。

ローランドエクスクルーシブに不可欠なチェックサム値は自動的に送信される。

本命令は楽器のメモリを直接操作するものなので、各自の責任の下で使用すること。

例

[ROLAND_EXCLUSIVE $40,$00,$7f,$00]

(ローランドエクスクルーシブとして$40,$00,$7f,$00をMIDI楽器へ送信)

MIDI　ヤマハエクスクルーシブ
[YAMAHA_EXCLUSIVE n1,n2,…]

・ni:0-127

ヤマハ系の楽器にエクスクルーシブメッセージ送信を行う

後述の[INSTRUMENT_ID]でメーカーIDをヤマハに設定しておくことが前提。

データの転送方式は「バルク・ダンプ」が使用される。

データはいくつでも記述可能。各値の設定範囲は0≦ni≦127であるが、128以上の値を設定した場合は7ビットずつに分割されて送信される。

(一般的なMIDI規格の慣例に従い、下位、上位の順に送信される。例:8192→$00,40)

また、"～"で括ればその文字コードを送信できる。ただし、同機能の共通コマンド.YAMAHA_EXCLUSIVEとは異なり複数行にパラメータを分けて書くことはできない。

ヤマハエクスクルーシブに不可欠な転送バイト数、チェックサム値は自動的に送信される。

本命令は楽器のメモリを直接操作するものなので、各自の責任の下で使用すること。

例

[YAMAHA_EXCLUSIVE $00,$00,$7e,$00]

(ヤマハエクスクルーシブとして$00,$00,$7e,$00をMIDI楽器へ送信)

MIDI　MIDI楽器IDの登録
[INSTRUMENT_ID n1,n2,n3]

・n1:0～127
・n2:0～127
・n3:0～127

Z-MUSICにそのトラックを演奏する楽器のメーカーID、デバイスID、モデルIDを設定する(所有しているMIDI楽器の各値については楽器のマニュアルを参照)。これは特に設定しなくても通常のコマンドの動作にはまったく影響しない。

ただし、

[ROLAND_EXCLUSIVE]
[EFFECT]

に関しては楽器個別の動作をするため、これらのコマンドを使用する前には必ず設定しなければならない。

n1=メーカーID(例:ROLAND=$41, KORG=$42, YAMAHA=$43)
n2=デバイスID(各ユーザーが楽器に設定した値。特に設定した覚えのない場合はその楽器のデフォルトのデバイスIDを設定する)
n3=モデルID(例:MT-32=$16, ＳＣ５５=$42 or $45)

初期値なし。

例

[INSTRUMENT_ID $41,$10,$42] (ROLAND,DEV-ID=$10,SC-55に設定)

■SC-55/mkII(SC-155/CM-300/CM-500)専用命令

MIDI　SC-55初期化
[SC55_INIT id]
[SC55_RESET id]
[GS_INIT id]
[GS_RESET id]

・id:$00～$7f

SC-55の初期化を行う。

idはSC-55のデバイスIDで省略可能、省略時は$10。

例

[SC55_INIT $10]

MIDI　SC-55の各パートのボイスリザーブ
[SC55_V_RESERVE id {n1,n2,…,n16}]
[SC55_PARTIAL_RESERVE id {n1,n2,…,n16}]
[GS_V_RESERVE id {n1,n2,…,n16}]
[GS_PARTIAL_RESERVE id {n1,n2,…,n16}]

・id:$00～$7f
・n1～n16:0～127

SC-55の各パートのボイスリザーブを行う。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n16)は必ず16個設定しなくてはならない。またパラメータ各値の総和は音源の同時発音数以内に収める必要がある。n1～n16はそれぞれパート1～16のボイスリザーブ値に対応し、10番目のパラメータはリズムパートに対応する。

同機能の共通コマンド.GS_PARTIAL_RESERVEとは異なり複数行にパラメータを分けて書くことはできない。

(SC-55マニュアルp79, またはMEASURE3 zm_gs_partial_reserve()参照))

例

[SC55_PARTIAL_RESERVE $10 {2,4,2,2,0,0,0,0,0,5,0,0,0,0,0,0}]

MIDI　SC-55のリバーブパラメータの設定
[SC55_REVERB id {n1,n2,…,n7}]
[GS_REVERB id {n1,n2,…,n7}]

・id:$00～$7f
・n1～n7:0-127

SC-55のリバーブパラメータの設定を行う。

idはSC-55のデバイスIDで省略可能、省略時は$10。

パラメータ(n1～n7)は7個まで任意の個数設定可能。同機能の共通コマンド.GS_REVERBとは異なり複数行にパラメータを分けて書くことはできない。

n1:REVERB MACRO	(00:Room1,01:Room2,02:Room3,03:Hall1,04:Hall2,05:Plate,06:Delay,07:Panning Delay(FB))
n2:REVERB CHARACTER	($00～$07)
n3:REVERB PRE-LPF	($00～$07)
n4:REVERB LEVEL	($00～$7F)
n5:REVERB TIME	($00～$7F)
n6:REVERB DELAY FEEDBACK	($00～$7F)
n7:REVERB SEND LEVEL TO CHORUS	($00～$7F)

(SC-55マニュアルp79参照)

例

[SC55_REVERB $10 ={4,4,0,88,70,0,0}]

MIDI　SC-55のコーラスパラメータの設定
[SC55_CHORUS id {n1,n2,…,n8}]

[GS_CHORUS id {n1,n2,…,n8}]

・id:$00〜$7f
・n1〜n8:0-127

SC-55のコーラスパラメータの設定を行う。
idはSC-55のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n8)は8個まで任意の個数設定可能。同機能の共通コマンド.GS_CHORUSとは異なり複数行にパラメータを分けて書くことはできない。

n1:CHORUS MACRO (00:Chorus1,01:Chorus2,02:Chorus3, 03:Chorus4,04:Feedback Chorus, 05:Flanger,06:Short Delay 07:Short Delay(FB))
n2:CHORUS PRE-LPF ($00〜$07)
n3:CHORUS LEVEL ($00〜$7F)
n4:CHORUS FEEDBACK ($00〜$7F)
n5:CHORUS DELAY ($00〜$7F)
n6:CHORUS RATE ($00〜$7F)
n7:CHORUS DEPTH ($00〜$7F)
n8:CHORUS SEND LEVEL TO REVERB ($00〜$7F)

(SC-55マニュアルp79参照)

例

[SC55_CHORUS $10 ={2,0,100,8,80,3,19,0}]

MIDI　SC-55のパートのパラメータを設定

[SC55_PART_SETUP pt,id {n1,n2,…,n119}]
[GS_PART_SETUP pt,id {n1,n2,…,n119}]
[SC-55_PART_PARAMETER pt,id {n1,n2,…,n119}]
[GS_PART_PARAMETER pt,id {n1,n2,…,n119}]

・pt:1〜16
・id:0-127
・n1〜n119:0-127

SC-55のパートのパラメータを設定する。
ptはパートナンバーを表し、1≦pt≦16で省略は不可。またパートナンバー=10はリズムパートに対応する。
idはSC-55のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n119)は119個まで任意の個数設定可能。ただし、同機能の共通コマンド.GS_PART_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:Rx CHANNEL (1〜16,17=OFF)
n2:Rx PITCH BEND (0〜1:OFF/ON)
n3:Rx CH PRESSURE(CAf) (0〜1:OFF/ON)
n4:Rx PROGRAM CHANGE (0〜1:OFF/ON)
n5:Rx CONTROL CHANGE (0〜1:OFF/ON)
n6:Rx POLY PRESSURE(PAf) (0〜1:OFF/ON)
n7:Rx NOTE MESSAGE (0〜1:OFF/ON)
n8:Rx RPN (0〜1:OFF/ON)
n9:Rx NRPN (0〜1:OFF/ON)
n10:Rx MODURATION (0〜1:OFF/ON)
n11:Rx VOLUME (0〜1:OFF/ON)
n12:Rx PANPOT (0〜1:OFF/ON)
n13:Rx EXPRESSION (0〜1:OFF/ON)
n14:Rx HOLD1 (0〜1:OFF/ON)
n15:Rx PORTAMENTO (0〜1:OFF/ON)
n16:Rx SOSTENUTO (0〜1:OFF/ON)
n17:Rx SOFT (0〜1:OFF/ON)
n18:MONO/POLY MODE (0〜1:MONO/POLY)
n19:ASSIGN MODE 0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
n20:USE FOR RHYTHM PART (0:OFF,1:MAP1,2:MAP2)
n21:PITCH KEY SHIFT ($28〜$58:−24〜+24[semitone])
n22:PITCH OFFSET FINE(upper) ($08〜$F8:−12.0〜+12.0[Hz])
n23:PITCH OFFSET FINE(lower) 〃
n24:PART LEVEL (0〜127)
n25:VELOCITY SENSE DEPTH (0〜127)
n26:VELOCITY SENSE OFFSET (0〜127)
n27:PART PANPOT (0:RANDOM,01(LEFT)〜64〜127(RIGHT))
n28:KEY RANGE LOW (0〜127:C-1〜G9)
n29:KEY RANGE HIGH (0〜127:C-1〜G9)
n30:CC1 CONTROL NUMBER (0〜127)
n31:CC2 CONTROL NUMBER (0〜127)
n32:CHORUS SEND DEPTH (0〜127)
n33:REVERB SEND DEPTH (0〜127)

n34:Vibrato rate ($0E〜$72:−50〜+50)
n35:Vibrato depth ($0E〜$72:−50〜+50)
n36:TVF cutoff freq. ($0E〜$50:−50〜+16)
n37:TVF resonance ($0E〜$72:−50〜+50)
n38:TVF&TVA Env.attack ($0E〜$72:−50〜+50)
n39:TVF&TVA Env.decay ($0E〜$72:−50〜+50)
n40:TVF&TVA Env.release ($0E〜$72:−50〜+50)
n41:Vibrato delay ($0E〜$72:−50〜+50)

n42:SCALE TUNING C ($00〜$7F:−64〜+63[cent])
n43:SCALE TUNING C# ($00〜$7F:−64〜+63[cent])
n44:SCALE TUNING D ($00〜$7F:−64〜+63[cent])
n45:SCALE TUNING D# ($00〜$7F:−64〜+63[cent])
n46:SCALE TUNING E ($00〜$7F:−64〜+63[cent])
n47:SCALE TUNING F ($00〜$7F:−64〜+63[cent])
n48:SCALE TUNING F# ($00〜$7F:−64〜+63[cent])
n49:SCALE TUNING G ($00〜$7F:−64〜+63[cent])
n50:SCALE TUNING G# ($00〜$7F:−64〜+63[cent])
n51:SCALE TUNING A ($00〜$7F:−64〜+63[cent])
n52:SCALE TUNING A# ($00〜$7F:−64〜+63[cent])
n53:SCALE TUNING B ($00〜$7F:−64〜+63[cent])

n54:MOD PITCH CONTROL ($28〜$58:−24〜+24[semitone])
n55:MOD TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n56:MOD AMPLITUDE CONTROL ($00〜$7F:−100.0〜+100.0[%])
n57:MOD LFO1 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n58:MOD LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
n59:MOD LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
n60:MOD LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
n61:MOD LFO2 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n62:MOD LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
n63:MOD LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
n64:MOD LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

n65:BEND PITCH CONTROL ($28〜$58:−24〜+24[semitone])
n66:BEND TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n67:BEND AMPLITUDE CONTROL ($00〜$7F:−100.0〜+100.0[%])
n68:BEND LFO1 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n69:BEND LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
n70:BEND LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
n71:BEND LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
n72:BEND LFO2 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n73:BEND LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
n74:BEND LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
n75:BEND LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

n76:CAf PITCH CONTROL ($28〜$58:−24〜+24[semitone])
n77:CAf TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n78:CAf AMPLITUDE CONTROL ($00〜$7F:−100.0〜+100.0[%])
n79:CAf LFO1 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n80:CAf LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
n81:CAf LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
n82:CAf LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
n83:CAf LFO2 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n84:CAf LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
n85:CAf LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
n86:CAf LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

n87:PAf PITCH CONTROL ($28〜$58:−24〜+24[semitone])
n88:PAf TVF CUTOFF CONTROL ($00〜$7F:−9600〜+9600[CENT])
n89:PAf AMPLITUDE CONTROL ($00〜$7F:−100.0〜+100.0[%])
n90:PAf LFO1 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n91:PAf LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
n92:PAf LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
n93:PAf LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
n94:PAf LFO2 RATE CONTROL ($00〜$7F:−10.0〜+10.0[Hz])
n95:PAf LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
n96:PAf LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])

n97:PAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n98:CC1 PITCH CONTROL ($28～$58:－24～＋24[semitone])
n99:CC1 TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n100:CC1 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n101:CC1 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n102:CC1 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n103:CC1 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n104:CC1 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n105:CC1 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n106:CC1 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n107:CC1 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n108:CC1 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

n109:CC2 PITCH CONTROL ($28～$58:－24～＋24[semitone])
n110:CC2 TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
n111:CC2 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
n112:CC2 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n113:CC2 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
n114:CC2 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
n115:CC2 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
n116:CC2 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
n117:CC2 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
n118:CC2 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
n119:CC2 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

(SC-55マニュアルp75,p79～80参照)

例

[SC55_PART_SETUP 16 {17,1,1,1}]

MIDI SC-55のドラムセットの設定
[SC55_DRUM_SETUP map,key,id {n1,n2,…,n8}]
[GS_DRUM_SETUP map,key,id {n1,n2,…,n8}]

・map:0,1
・key:0-127
・id:0-127
・n1～n8:0-127

SC-55のドラムセットの設定を変える。
mapはマップナンバーを表し設定範囲は0,1。
keyはノートナンバーを表し設定範囲は0≦key≦127。
idはSC-55のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n8)は8個まで任意の個数設定可能。同機能の共通コマンド.GS_DRUM_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:PLAY KEY NUMBER (0～127)
n2:LEVEL (0～127)
n3:ASSIGN GROUP NUMBER (0:NON,1～127)
n4:PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH (0～127:0.0～1.0)
n6:CHORUS DEPTH (0～127:0.0～1.0)
n7:Rx NOTE OFF (0～1:OFF/ON)
n8:Rx NOTE ON (0～1:OFF/ON)

(SC-55マニュアルp82参照)

例

[GS_DRUM_SETUP 0,24,$10 {24,127,0,0,10,15,0,1}]

MIDI SC-55のドラムセット名の設定
[SC55_DRUM_NAME map,id str]
[GS_DRUM_NAME map,id str]

・map:0,1
・id:0-127
・str:文字列

SC-55のドラムセット名の設定を行う。
mapはマップナンバーを表し設定範囲は0,1。
idはSC-55のデバイスIDで省略可能、省略時は$10。
strはドラセット名を表す文字列で、"～"で括り12文字以内で指定する。
(SC-55マニュアルp82参照)

例

[GS_DRUM_NAME 0,$10 "USER SET"]

MIDI SC-55コンソールへの文字列表示
[SC55_PRINT id,str]
[GS_PRINT id,str]

・id:0-127
・str:文字列

SC-55のコンソールに文字列を表示する。
文字列strは"～"で括り32文字以内に収めなければならない。
idはSC-55のデバイスIDで省略可能、省略時は$10。

例

[GS_PRINT $10,"HELLO"]

MIDI SC-55グラフィックディスプレイのドットパターン表示
[SC55_DISPLAY id {n1,n2,…,n16}]
[GS_DISPLAY id {n1,n2,…,n16}]

・id:0-127
・n1～n16:%0000000000000000～%1111111111111111

SC-55のグラフィックディスプレイにドットパターン(16x16)を表示する。
idはSC-55のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n16)は必ず16個。同機能の共通コマンド.GS_DISPLAYとは異なり複数行にパラメータを分けて書くことはできない。

n1:1段目のドットパターン
n2:2段目のドットパターン
:
:
n16:16段目のドットパターン

例

[GS_DISPLAY $10 {$ffff,9,8,7,65,40,0,0,0,1,2,3,4,5,$8000,$ffff}]

■GM音源専用命令

MIDI GM音源初期化
[GM_SYSTEM_ON]

GM音源を初期化状態にリセットする。パラメータはなし。

例

[GM_SYSTEM_ON]

■SC-88/SC-88VL専用命令

MIDI SC-88モード設定
[SC88_MODE_SET md,id]

・md:0,1
・id:0-127

SC-88のモードの設定を行う。
idはSC-88のデバイスIDで省略可能、省略時は$10。
mdはモード値を表し、md=0でSingleモード、md=1でDoubleモードが設定される。(SC-88VLマニュアルp77参照)

例

[SC88_MODE_SET 1,$10]

MIDI SC-88リバーブパラメータの設定
[SC88_REVERB id {n1,n2,...,n8}]

・id:0-127
・n1～n8:下記参照

SC-88のリバーブパラメータの設定を行う。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n8)は8個以内で任意の個数設定可能。同機能の共通コマンド.SC88_REVERBとは異なり複数行にパラメータを分けて書くことはできない。

n1:REVERB MACRO (00:Room1,01:Room2,
02:Room3,03:Hall1,04:Hall2
05:Plate,06:Delay,
07:Panning Delay(FB))
n2:REVERB CHARACTER ($00～$07)
n3:REVERB PRE-LPF ($00～$07)
n4:REVERB LEVEL ($00～$7F)
n5:REVERB TIME ($00～$7F)
n6:REVERB DELAY FEEDBACK ($00～$7F)
n7:REVERB SEND LEVEL TO CHORUS ($00～$7F/SC-88ではダミー)
n8:REVERB PRE-DELAY TIME ($00～$7F)
(SC-88VLマニュアルp77参照)

例

[SC88_REVERB $10 {0,0,127,10,10,0,127}]

MIDI　SC-88コーラスパラメータの設定
[SC88_CHORUS id {n1,n2,...,n9}]

・id:0-127
・n1～n9:下記参照

SC-88のコーラスパラメータの設定を行う。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n9)は9個以内で任意の個数設定可能。同機能の共通コマンド.SC88_CHORUSとは異なり複数行にパラメータを分けて書くことはできない。

n1:CHORUS MACRO	(00:Chorus1,01:Chorus2, 02:Chorus3,03:Chorus4 04:Feedback Chorus, 05:Flanger,06:Short Delay 07:Short Delay(FB))
n2:CHORUS PRE-LPF	($00～$07)
n3:CHORUS LEVEL	($00～$7F)
n4:CHORUS FEEDBACK	($00～$7F)
n5:CHORUS DELAY	($00～$7F)
n6:CHORUS RATE	($00～$7F)
n7:CHORUS DEPTH	($00～$7F)
n8:CHORUS SEND LEVEL TO REVERB	($00～$7F)
n9:CHORUS SEND LEVEL TO DELAY	($00～$7F)

(SC-88VLマニュアルp78参照)

例

[SC88_CHORUS $10 {0,3,60,10,10,10,24,30,30}]

MIDI　SC-88ディレイパラメータの設定
[SC88_DELAY id {n1,n2,...,n11}]

・id:0-127
・n1～n11:下記参照

SC-88のディレイパラメータの設定を行う。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n11)は11個以内で任意の個数設定可能。同機能の共通コマンド.SC88_DELAYとは異なり複数行にパラメータを分けて書くことはできない。

n1:DELAY MACRO	(00:Delay1,01:Delay2,02:Delay3, 03:Delay4,04:Pan Delay1, 05:Pan Delay2,06:Pan Delay3, 07:Pan Delay4,08:Delay to Reverb, 09:Pan Repeat)
n2:DELAY PRE-LPF	($00～$07)
n3:DELAY TIME CENTER	($01～$73:0.1ms～sec)
n4:DELAY TIME RATIO LEFT	($01～$78:4～500%)
n5:DELAY TIME RATIO RIGHT	($01～$78:4～500%)
n6:DELAY LEVEL CENTER	($00～$7F)
n7:DELAY LEVEL LEFT	($00～$7F)
n8:DELAY LEVEL RIGHT	($00～$7F)
n9:DELAY LEVEL	($00～$7F)
n10:DELAY FEEDBACK	($00～$7F:-64～+63)
n11:DELAY SENDLEVEL TO REVERB	($00～$7F)

(SC-88VLマニュアルp78参照)

例

[SC88_DELAY $10 {0,3,10,22,22,64,64,64,64,64,30}]

MIDI　SC-88イコライザパラメータの設定
[SC88_EQUALIZER id {n1,n2,n3,n4}]

・id:0-127
・n1～n4:下記参照

SC-88のイコライザパラメータの設定を行う。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n4)は4個以内で任意の個数設定可能。同機能の共通コマンド.SC88_EQUALIZERとは異なり複数行にパラメータを分けて書くことはできない。

n1:EQ LOW FREQ	(0,1:200Hz,400Hz)
n2:EQ LOW GAIN	($34～$4c:-12～+12dB)
n3:EQ HIGH FREQ	(0,1:3kHz,6kHz)
n4:EQ HIGH GAIN	($34～$4c:-12～+12dB)

(SC-88VLマニュアルp78参照)

例

[SC88_EQUALIZER $10 {0,$40,1,$40}]

MIDI　SC-88パートパラメータの設定
[SC88_PART_SETUP pt,id {n1,n2,...,n127}]
[SC88_PART_PARAMETER pt,id {n1,n2,...,n127}]

・pt:1～16
・id:0-127
・n1～n127:下記参照

SC-88パートパラメータの設定する。
ptはパートナンバーを表し、1≦pt≦16で省略は不可。またパートナンバー＝10はリズムパートに対応する。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n127)は127個まで任意の個数設定可能。ただし、同機能の共通コマンド.SC88_PART_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:Rx CHANNEL	(1～16,17＝OFF)
n2:Rx PITCH BEND	(0～1:OFF/ON)
n3:Rx CH PRESSURE(CAf)	(0～1:OFF/ON)
n4:Rx PROGRAM CHANGE	(0～1:OFF/ON)
n5:Rx CONTROL CHANGE	(0～1:OFF/ON)
n6:Rx POLY PRESSURE(PAf)	(0～1:OFF/ON)
n7:Rx NOTE MESSAGE	(0～1:OFF/ON)
n8:Rx RPN	(0～1:OFF/ON)
n9:Rx NRPN	(0～1:OFF/ON)
n10:Rx MODURATION	(0～1:OFF/ON)
n11:Rx VOLUME	(0～1:OFF/ON)
n12:Rx PANPOT	(0～1:OFF/ON)
n13:Rx EXPRESSION	(0～1:OFF/ON)
n14:Rx HOLD1	(0～1:OFF/ON)
n15:Rx PORTAMENTO	(0～1:OFF/ON)
n16:Rx SOSTENUTO	(0～1:OFF/ON)
n17:Rx SOFT	(0～1:OFF/ON)
n18:MONO/POLY MODE	(0～1:MONO/POLY)
n19:ASSIGN MODE	(0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
n20:USE FOR RHYTHM PART	(0:OFF,1:MAP1,2:MAP2)
n21:PITCH KEY SHIFT	($28～$58:－24～＋24[semitone])
n22:PITCH OFFSET FINE(upper)	($08～$F8:－12.0～＋12.0[Hz])
n23:PITCH OFFSET FINE(lower)	〃
n24:PART LEVEL	(0～127)
n25:VELOCITY SENSE DEPTH	(0～127)
n26:VELOCITY SENSE OFFSET	(0～127)
n27:PART PANPOT	(0:RANDOM,01(LEFT)～64～127(RIGHT))
n28:KEY RANGE LOW	(0～127:C-1～G9)
n29:KEY RANGE HIGH	(0～127:C-1～G9)
n30:CC1 CONTROL NUMBER	(0～127)
n31:CC2 CONTROL NUMBER	(0～127)
n32:CHORUS SEND LEVEL	(0～127)
n33:REVERB SEND LEVEL	(0～127)
n34:Rx BANK SELECT	(0:OFF,1:ON)
n35:Rx BANK SELECT LSB	(0:OFF,1:ON)
n36:PITCH FINE TUNE(Upper)	($00 00～$40 00～$7f 7f:-100～0～+100)
n37:PITCH FINE TUNE(Lower)	
n38:DELAY SEND LEVEL	(0～127)
n39:Vibrato rate	($00～$7f:－64～＋63)
n40:Vibrato depth	($00～$7f:－64～＋63)
n41:TVF cutoff freq.	($00～$7f:－64～＋63)
n42:TVF resonance	($00～$7f:－64～＋63)
n43:TVF&TVA Env.attack	($00～$7f:－64～＋63)
n44:TVF&TVA Env.decay	($00～$7f:－64～＋63)
n45:TVF&TVA Env.release	($00～$7f:－64～＋63)
n46:Vibrato delay	($00～$7f:－64～＋63)
n47:SCALE TUNING C	($00～$7F:－64～＋63[cent])
n48:SCALE TUNING C#	($00～$7F:－64～＋63[cent])
n49:SCALE TUNING D	($00～$7F:－64～＋63[cent])

n50:SCALE TUNING D#　($00～$7F:－64～＋63[cent])
n51:SCALE TUNING E　($00～$7F:－64～＋63[cent])
n52:SCALE TUNING F　($00～$7F:－64～＋63[cent])
n53:SCALE TUNING F#　($00～$7F:－64～＋63[cent])
n54:SCALE TUNING G　($00～$7F:－64～＋63[cent])
n55:SCALE TUNING G#　($00～$7F:－64～＋63[cent])
n56:SCALE TUNING A　($00～$7F:－64～＋63[cent])
n57:SCALE TUNING A#　($00～$7F:－64～＋63[cent])
n58:SCALE TUNING B　($00～$7F:－64～＋63[cent])

n59:MOD PITCH CONTROL　($28～$58:－24～＋24[semitone])
n60:MOD TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n61:MOD AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n62:MOD LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n63:MOD LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n64:MOD LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n65:MOD LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n66:MOD LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n67:MOD LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n68:MOD LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n69:MOD LFO2 TVA DEPTH　($00～$7F:0～100.0[%])

n70:BEND PITCH CONTROL　($40～$58:－24～＋24[semitone])
n71:BEND TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n72:BEND AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n73:BEND LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n74:BEND LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n75:BEND LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n76:BEND LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n77:BEND LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n78:BEND LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n79:BEND LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n80:BEND LFO2 TVA DEPTH　($00～$7F:0～100.0[%])

n81:CAf PITCH CONTROL　($28～$58:－24～＋24[semitone])
n82:CAf TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n83:CAf AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n84:CAf LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n85:CAf LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n86:CAf LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n87:CAf LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n88:CAf LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n89:CAf LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n90:CAf LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n91:CAf LFO2 TVA DEPTH　($00～$7F:0～100.0[%])

n92:PAf PITCH CONTROL　($28～$58:－24～＋24[semitone])
n93:PAf TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n94:PAf AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n95:PAf LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n96:PAf LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n97:PAf LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n98:PAf LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n99:PAf LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n100:PAf LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n101:PAf LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n102:PAf LFO2 TVA DEPTH　($00～$7F:0～100.0[%])

n103:CC1 PITCH CONTROL　($28～$58:－24～＋24[semitone])
n104:CC1 TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n105:CC1 AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n106:CC1 LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n107:CC1 LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n108:CC1 LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n109:CC1 LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n110:CC1 LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n111:CC1 LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n112:CC1 LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n113:CC1 LFO2 TVA DEPTH　($00～$7F:0～100.0[%])
n114:CC2 PITCH CONTROL　($28～$58:－24～＋24[semitone])
n115:CC2 TVF CUTOFF CONTROL　($00～$7F:－9600～＋9600[CENT])
n116:CC2 AMPLITUDE CONTROL　($00～$7F:－100.0～＋100.0[%])
n117:CC2 LFO1 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n118:CC2 LFO1 PITCH DEPTH　($00～$7F:0～600[cent])
n119:CC2 LFO1 TVF DEPTH　($00～$7F:0～2400[cent])
n120:CC2 LFO1 TVA DEPTH　($00～$7F:0～100.0[%])
n121:CC2 LFO2 RATE CONTROL　($00～$7F:－10.0～＋10.0[Hz])
n122:CC2 LFO2 PITCH DEPTH　($00～$7F:0～600[cent])
n123:CC2 LFO2 TVF DEPTH　($00～$7F:0～2400[cent])
n124:CC2 LFO2 TVA DEPTH　($00～$7F:0～100.0[%])

n125:TONE MAP NUMBER　(0:SELECTED,1:SC-55,2:SC-88)
n126:TONE MAPO NUMBER　(1:SC-55,2:SC-88)
n127:EQ ON/OFF　(0:OFF,1:ON)
(SC-88VLマニュアルp79～p81参照)

例

[SC88_PART_SETUP 16 {17,1,1,1}]

MIDI　SC-88のドラムセットアップ設定
[SC88_DRUM_SETUP map,key,id {n1,n2,...,n9}]

・map:0,1
・key:0-127
・id:0-127
・n1～n8:0-127

SC-88のドラムセットアップの設定を行う。
mapはマップナンバーを表し設定範囲は0,1。
keyはノートナンバーを表し設定範囲は0≦key≦127。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n9)は9個まで任意の個数設定可能。同機能の共通コマンド.SC88_DRUM_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:PLAY KEY NUMBER　(0～127)
n2:LEVEL　(0～127)
n3:ASSIGN GROUP NUMBER　(0:NON,1～127)
n4:PANPOT　(0:RANDOM,0(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH　(0～127:0.0～1.0)
n6:CHORUS DEPTH　(0～127:0.0～1.0)
n7:Rx NOTE OFF　(0～1:OFF/ON)
n8:Rx NOTE ON　(0～1:OFF/ON)
n9:DELAY SEND LEVEL　(0-127:0.0～1.0)
(SC-88VLマニュアルp82参照)

例

[SC88_DRUM_SETUP $10 {42,127,0,64,64,64,1,1,24}]

MIDI　SC-88のドラムセット名の設定
[SC88_DRUM_NAME map,id str]

・map:0,1　または USER65,USER66
・id:0-127
・str:文字列

SC-88のドラムセット名の設定を行う。
mapはマップナンバーを表し設定範囲は0,1。ユーザードラムセットの名前を設定する場合にはUSER65,またはUSER66という文字列を与えて指定する。
idはSC-88のデバイスIDで省略可能、省略時は$10。
strはドラムセット名を表す文字列で、"～"で括り12文字以内で指定する。
(SC-88VLマニュアルp82参照)

例

[SC88_DRUM_NAME USER66,$10 "USER SET"]

MIDI　SC-88ユーザー音色の設定
[SC88_USER_INST bk,pgm,id {n1,n2,...,n11}]

・bk:0-1
・pgm:1-128
・id:0-127
・n1～n11:下記参照

SC-88ユーザー音色の設定を行う。
bkは設定先音色バンク番号を表し、設定範囲は0,1。bk=0がバリエーション番号64,bk=1がバリエーション番号65に対応する。

pgmは設定先音色番号で、設定範囲は1～128。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n11)は11個まで任意の個数設定可能。同機能の共通コマンド.SC88_USER_INSTとは異なり複数行にパラメータを分けて書くことはできない。

n1:SOURCE TONE#(MAP)	(1,2)
n2:BANK NUMBER MSB	(0～127)
n3:PROGRAM NUMBER	(1～128)
n4:Vibrato rate	($00～$7f:－64～＋63)
n5:Vibrato depth	($00～$7f:－64～＋63)
n6:TVF cutoff freq.	($00～$7f:－64～＋63)
n7:TVF resonance	($00～$7f:－64～＋63)
n8:TVF&TVA Env.attack	($00～$7f:－64～＋63)
n9:TVF&TVA Env.decay	($00～$7f:－64～＋63)
n10:TVF&TVA Env.release	($00～$7f:－64～＋63)
n11:Vibrato delay	($00～$7f:－64～＋63)

(SC-88VLマニュアルp82参照)

例

```
[SC88_USER_INST 0,4,$10 {1,0,10,64,64,64,64,64,64,64,64}]
```

MIDI　SC-88のユーザードラムセットの設定
[SC88_USER_DRUM st,nt,id {n1,n2,...,n12}]

・st:0-1
・nt:0-127
・id:0-127
・n1～n12:下記参照

stは設定先ドラムセット番号で設定範囲は0,1。st=0がドラムセット番号65、st=1がドラムセット番号66に対応する。
ntは登録先ノート番号で設定範囲は0-127。
idはSC-88のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n12)は12個まで任意の個数設定可能。同機能の共通コマンド.SC88_DRUM_INSTとは異なり複数行にパラメータを分けて書くことはできない。

n1:PLAY KEY NUMBER	(0～127)
n2:LEVEL	(0～127)
n3:ASSIGN GROUP NUMBER	(0:NON,1～127)
n4:PANPOT	(0:RANDOM,01(LEFT)～64～127(RIGHT))
n5:REVERB DEPTH	(0～127:0.0～1.0)
n6:CHORUS DEPTH	(0～127:0.0～1.0)
n7:Rx NOTE OFF	(0～1:OFF/ON)
n8:Rx NOTE ON	(0～1:OFF/ON)
n9:DELAY SEND LEVEL	(0-127:0.0～1.0)
n10:SOURCE DRUM SET#(MAP)	(1-2)
n11:PROGRAM NUMBER	(0-127)
n12:SOURCE NOTE NUMBER	(0-127)

(SC-88VLマニュアルp82参照)

例

```
[SC88_USER_DRUM 0,32,$10 {24,127,0,64,10,10,0,1,10,1,10,24}]
```

■MT-32/CM-32L/CM-64/CM-500専用命令

MIDI　MT-32の初期化
[MT32_INIT id]
[MT32_RESET id]

・id:0-127
MT-32の初期化を行う。
idはMT-32のデバイスIDで省略可能、省略時は$10。

例

```
[MT32_INIT $10]
```

MIDI　MT-32の各パートのパーシャルリザーブ
[MT32_P_RESERVE id {n1,n2,…,n9}]
[MT32_PARTIAL_RESERVE id {n1,n2,…,n9}]

・id:0-127
・n1～n9:0-127
MT-32の各パートのパーシャルリザーブを行う。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n9)は必ず9個なければならない。パラメータ各値の総和は32以内。
n1～n8はパート1～8のパーシャルリザーブ値に対応し、n9はリズムパートに対応する。同機能の共通コマンド.MT32_PARTIAL_RESERVEとは異なり複数行にパラメータを分けて書くことはできない。

例

```
[MT32_PARTIAL_RESERVE $10 {2,2,2,2,0,0,1,2,4}]
```

MIDI　MT-32リバーブパラメータの設定
[MT32_REVERB id {n1,n2,n3}]

id:0-127
n1～n3:0-127
MT-32のリバーブパラメータを設定する。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n3)は3個まで任意の個数設定可能。同機能の共通コマンド.MT32_REVERBとは異なり複数行にパラメータを分けて書くことはできない。

n1:REVERB MODE	(0:Room,1:Hall,2:Plate,3:Tap Delay)
n2:REVERB TIME	(0～7)
n3:REVERB LEVEL	(0～7)

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
[MT32_REVERB $10 {3,3,5}]
```

MIDI　MT-32各パートのMIDIチャンネル設定
[MT32_PART_SETUP id {n1,n2,…,n9}]

・id:0-127
・n1～n9:0-127
MT-32の各パートのMIDIチャンネルを設定する。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n9)は9個まで任意の個数設定可能。
n1～n8はパート1～8のMIDIチャンネル値に対応し、n9はリズムパートに対応する。
同機能の共通コマンド.MT32_PART_SETUPとは異なり複数行にパラメータを分けて書くことはできない。
パラメータ(n1～n9)はMIDIチャンネル値なので1～16まで有効。ただし、0あるいは17以上を与えた場合はそのパートを無効(OFF)と見なす。
(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
[MT32_PART_SETUP $10 {2,3,4,5,6,7,8,9,10}]
```

MIDI　MT-32リズムキットの設定
[MT32_DRUM_SETUP n,id {n1,n2,n3,n4}]
[MT32_RHYTHM_SETUP n,id {n1,n2,n3,n4}]

・n:24～87
・id:0-127
MT-32のリズムキットの設定を変更する。
nは変更対象ノートナンバーを表し、設定範囲は24≦n≦87。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n4)は4個まで任意の個数設定可能。同機能の共通コマンド.MT32_DRUM_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:TIMBRE	(0～63:i1～64,64～127:r1～64)
n2:OUTPUT LEVEL	(0～100)
n3:PANPOT	(0～14)
n4:REVERB SWITCH	(0～1:OFF/ON)

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

```
[MT32_DRUM_SETUP 24,$10 {10,100,12,1}]
```

MIDI　MT-32コモンパラメータの設定
[MT32_COMMON n,id {str,n1,n2,n3,n4}]

・n:1-64
・id:0-127
・str:文字列
・n1～n4:0-127
MT-32の音色のコモンパラメータを設定する。
nは設定対象ティンバー番号で、設定範囲は1～64。
idはMT-32のデバイスIDで省略可能、省略時は$10。
strは音色名で、"～"で括り、10文字以内(省略不可)の文字列で指定する。
パラメータ(n1～n4)は4個まで任意の個数設定可能。同機能の共通コマンド.MT32_COMMONとは異なり複数行にパラメータを分けて書くことはできない。

n1:Structure of Partial#1&2 (0～12:1～13)
n2:Structure of Partial#3&4 (0～12:1～13)
n3:PARTIAL MUTE (0～15)
n4:ENV MODE (0～1:NORMAL,NO SUSTAIN)

(MT-32マニュアルp34, CM-64マニュアルp29参照)

例

[MT32_COMMON 1,$10 {1,1,0,0}]

MIDI MT-32パーシャルパラメータの設定
[MT32_PARTIAL n,p,id {n1,n2,…,n58}]

・n:1-64
・p:1-4
・id:0-127
・n1～n58:0-127

MT-32の音色のパーシャルパラメータを設定する。
nは設定対象ティンバー番号で設定範囲は1≦n≦64。
pは設定対象パーシャルナンバーで設定範囲は1≦p≦4。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n58)は58個まで任意の個数設定可能。同機能の共通コマンド.MT32_PARTIALとは異なり複数行にパラメータを分けて書くことはできない。

n1:WG PITCH COARSE (0～96:C1,C#1,～,C9)
n2:WG PITCH FINE (0～100:－50～＋50)
n3:WG PITCH KEYFOLLOW (0～16:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2,s1,s2)
n4:WG PITCH BENDER SW (0～1:OFF/ON)
n5:WG WAVEFORM/PCM BANK (0～3:SQU1,SAW1,SQU2,SAW2)
n6:WG PCM WAVE (0～127:1～128)
n7:WG PULSE WIDTH (0～100)
n8:WG PW VELO SENS (0～14:－7～＋7)
n9:P-ENV DEPTH (0～10)
n10:P-ENV VELO SENS (0～3)
n11:P-ENV TIME KEYF (0～4)
n12:P-ENV TIME 1 (0～100)
n13:P-ENV TIME 2 (0～100)
n14:P-ENV TIME 3 (0～100)
n15:P-ENV TIME 4 (0～100)
n16:P-ENV LEVEL 0 (0～100:-50～+50)
n17:P-ENV LEVEL 1 (0～100:-50～+50))
n18:P-ENV LEVEL 2 (0～100:-50～+50)
n19:P-ENV SUSTAIN LEVEL (0～100:-50～+50)
n20:END LEVEL (0～100:-50～+50)
n21:P-LFO RATE (0～100)
n22:P-LFO DEPTH (0～100)
n23:P-LFO MOD SENS (0～100)
n24:TVF CUTOFF FREQ (0～100)
n25:TVF RESONANCE (0～30)
n26:TVF KEYFOLLOW (0～14:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2)
n27:TVF BIAS POINT/DIR (0～127)
n28:TVF BIAS LEVEL (0～14:－7～＋7)
n29:TVF ENV DEPTH (0～100)
n30:TVF ENV VELO SENS (0～100)
n31:TVF ENV DEPTH KEYF (0～4)
n32:TVF ENV TIME KEYF (0～4)
n33:TVF ENV TIME 1 (0～100)
n34:TVF ENV TIME 2 (0～100)
n35:TVF ENV TIME 3 (0～100)
n36:TVF ENV TIME 4 (0～100)
n37:TVF ENV TIME 5 (0～100)
n38:TVF ENV LEVEL 1 (0～100)
n39:TVF ENV LEVEL 2 (0～100)
n40:TVF ENV LEVEL 3 (0～100)
n41:TVF ENV SUSTAIN LEVEL (0～100)
n42:TVA LEVEL (0～100)
n43:TVA VELO SENS (0～100:－50～＋50
n44:TVA BIAS POINT 1 (0～127)
n45:TVA BIAS LEVEL 1 (0～12:－12～0)
n46:TVA BIAS POINT 2 (0～127)
n47:TVA BIAS LEVEL 2 (0～12:－12～0)
n48:TVA ENV TIME KEYF (0～4)
n49:TVA ENV TIME V_FOLLOW (0～4)
n50:TVA ENV TIME 1 (0～100)
n51:TVA ENV TIME 2 (0～100)
n52:TVA ENV TIME 3 (0～100)
n53:TVA ENV TIME 4 (0～100)
n54:TVA ENV TIME 5 (0～100)
n55:TVA ENV LEVEL 1 (0～100)
n56:TVA ENV LEVEL 2 (0～100)
n57:TVA ENV LEVEL 3 (0～100)
n58:TVA ENV SUSTAIN LEVEL (0～100)

(MT-32マニュアルp34, CM-64マニュアルp29参照)

例

[MT32_PARTIAL 1,1,$10 {64,50,11,1,1,127,50,7}]

MIDI MT-32パッチの設定
[MT32_PATCH n,id {n1,n2,…,n7}]

・n:1-128
・id:0-127
・n1～n7:0-127

MT-32のパッチを設定する。
nは設定対象パッチナンバーで、設定範囲は1≦n≦128。
idはMT-32のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n7)は7個まで任意の設定個数設定可能。同機能の共通コマンド.MT32_PATCHとは異なり複数行にパラメータを分けて書くことはできない。

n1:TIMBRE GROUP (0～3:a,b,i,r)
n2:TIMBRE NUMBER (0～63)
n3:KEY SHIFT (0～48:－24～＋24)
n4:FINE TUNE (0～100:－50～＋50)
n5:BENDER RANGE (0～24)
n6:ASSIGN MODE (0～3:POLY1～4)
n7:REVERB SWITCH (0～1:OFF/ON)

(MT-32マニュアルp35, CM-64マニュアルp30参照)

例

[MT32_PATCH 1,$10 {0,0,0,50,12,1,1}]

MIDI MT-32コンソール文字列表示
[MT32_PRINT id str]

・id:0-127
・str:文字列

MT-32のコンソールに文字列を表示する。
idはMT-32のデバイスIDで省略可能、省略時は$10。
strは文字列で、"～"でくくり20文字以内で指定する。
(MEASURE3 zm_mt32_print()参照)

例

[MT32_PRINT $10 "HELLO!"]

■U220/U20専用命令

MIDI U220セットアップパラメータの設定
[U220_SETUP id {n1,n2,…,n7}]

・id:0-127
・n1～n7:0-127

U220のセットアップパラメータを設定する。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1～n7)は必ず7個指定しなければならない。また同機能の共通コマンド.U220_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:Chorus SW (0～1:OFF/ON)
n2:Reverb SW (0～1:OFF/ON)
n3:Rx Cntrl Channel (1～17:1～16,OFF)
n4:Patch Change (0～5:MAP1～MAP4,Dir,OFF)
n5:Timbre Change (0～5:MAP1～MAP4,Dir,OFF)
n6:Rhythm Change (0～5:MAP1～MAP4,Dir,OFF)
n7:R.Inst Assign (0～5:MAP1～MAP4,Dir,OFF)

(U220マニュアルp46,p146,p148)

例

[U220_SETUP $10 {1,1,1,5,4,4,4}]

MIDI U220パートパラメータの設定

[U220_PART_SETUP pt,id {n1,n2,…,n13}]

・pt:1-6
・id:0-127
・n1〜n13:0-127

U220のテンポラリパッチのパートパラメータを設定する。
ptはパートナンバーを表し、設定範囲は1≦pt≦6。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n13)は必ず13個指定しなければならない。また同機能の共通コマンド.U220_PART_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

```
n1:Timbre Number        (1〜128)
n2:Voice Reserve        (0〜30)
n3:Receive Channel      (1〜17:1〜16,OFF)
n4:Key Range Low        (0〜127:C-1〜G9)
n5:Key Range High       (0〜127:C-1〜G9)
n6:Velo Level           (0〜1:Above/Below)
n7:Velo Threshold       (0〜127)
n8:Output Assign        (0〜4:Dry,Rev,Cho,Dir1,Dir2)
n9:Level                (0〜127)
n10:Pan                 (0〜15:L7〜M〜R7)
n11:Rx Volume           (0〜1:OFF/ON)
n12:Rx Pan              (0〜1:OFF/ON)
n13:Rx Hold             (0〜1:OFF/ON)
```

(U220マニュアルp58,p149参照)

例

```
[U220_PART_SETUP 1,$10 {1,3,1,0,127,1,127,0,0,8,0,0,0}]
```

MIDI U220コモンパラメータの設定

[U220_COMMON id {n1,n2,…,n18}]

・id:0-127
・n1〜n18:0-127

U220のテンポラリパッチのコモンパラメータを設定する。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n18)は必ず18個指定しなければならない。また同機能の共通コマンド.U220_COMMONとは異なり複数行にパラメータを分けて書くことはできない。

```
n1:Chorus Type          (0〜4:Chorus1,Chorus2,
                          FB-Chorus,Flanger,Short Delay)
n2:Chorus Out Mode      (0〜1:Pre Rev,Post Rev)
n3:Chorus Level         (0〜31)
n4:Chorus Delay         (0〜31)
n5:Chorus Rate          (0〜31)
n6:Chorus Depth         (0〜31)
n7:Chorus Feedback      (1〜63:-31〜0〜+31)
n8:Reverb Type          (0〜7:Room1〜3,Hall1〜2,
                          GateDelay,CrossDelay)
n9:Reverb Time          (0〜31)
n10:Reverb Level        (0〜31)
n11:Reverb Delay Feedback       (0〜31)
n12:Reverb Pre Delay Feedback   (0〜31)
n13:Param1 #            (0〜63:0〜5,7〜31,64〜95,OFF)
n14:Param1 Param        (0:Timbre Level,1:Env Attack
                          2:Env Decay,3:Env Sustain
                          4:Env Release,5:A.Bend Depth
                          6:A.Bend Rate,7:Detune Depth
                          8:Vib Rate,9:Vib Wave Form
                          10:Vib Depth,11:Vib Delay
                          12:VibRiseTime,13:VibMod Depth
                          14:Chorus Level,15:Chorus Rate
                          16:ChorusFeedback,17:ReverbLevel
                          18:Delay Feedback)
n15:Param2 #            (Same as Param1 #)
n16:Param2 Param        (Same as Param1 Param)
n17:Param3 #            (Same as Param1 #)
n18:Param3 Param        (Same as Param1 Param)
```

(U220マニュアルp54,p149参照)

例

```
[U220_COMMON $10 {0,1,20,20,20,20,32,0,2,2,2,2,63,0,63,0,63,0}]
```

MIDI U220音色パラメータ設定

[U220_TIMBRE n,id {str,n1,n2,…,n26}]

・n:1-128
・id:0-127
・str:文字列
・n1〜n26:0-127

U220に音色パラメータを設定する。
nは登録先音色番号を表し、設定範囲は1≦n≦128。
idはU220のデバイスIDで省略可能、省略時は$10。
strは音色名を表す文字列で、"〜"で括り、12文字以内で指定する。
パラメータ(n1〜n26)は必ず26個指定しなければならない。また同機能の共通コマンド.U220_TIMBREとは異なり複数行にパラメータを分けて書くことはできない。

```
n1:Tone Media           (0〜31:I,1〜31)
n2:Tone Number          (1〜128)
n3:Timbre Level         (0〜127)
n4:Velocity Sens        (1〜15:-7〜+7)
n5:Channel Press Sens   (1〜15:-7〜+7)
n6:Env Attack Rate      (1〜15:-7〜+7)
n7:Env Decay Rate       (1〜15:-7〜+7)
n8:Env Sustain Level    (1〜15:-7〜+7)
n9:Env Release Rate     (1〜15:-7〜+7)
n10:Pitch Shift Coarse  (8〜56:-24〜+24)
n11:Pitch Shift Fine    (14〜114:-50〜+50)
n12:Bend Range Lower    (0〜15:-36,-24,-12〜0)
n13:Bend Range Upper    (0〜12)
n14:Channel After Sens  (0〜27:-36,-24,-12〜+12)
n15:Poly After Sens     (0〜27:-36,-24,-12〜+12)
n16:Auto Bend Depth     (0〜27:-36,-24,-12〜+12)
n17:Auto Bend Rate      (0〜15)
n18:Detune Depth        (0〜15)
n19:Rate                (0〜63)
n20:Waveform            (0〜8)
n21:Depth               (0〜15)
n22:Delay               (0〜15)
n23:Rise Time           (0〜15)
n24:Modulation Depth    (0〜15)
n25:Ch After Sens       (0〜15)
n26:Poly After Sens     (0〜15)
```

(U220マニュアルp72,p149参照)

例

```
[U220_TIMBRE 1,$10 {"SYNTHE-2",0,1,127,8,8,8,8,8,8,32,64,2,2,
                     14,14,14,0,0,0,0,0,0,0,0,0,0}]
```

(本来はコマンドライン途中で改行することはできないがレイアウトの都合上改行している)

MIDI U220ドラムパラメータの設定

[U220_DRUM_SETUP id {n1,n2,…,n7}]

・id:0-127
・n1〜n7:0-127

U220のテンポラリパッチのドラムパラメータを設定する。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n7)は必ず7個指定しなければならない。また同機能の共通コマンド.U220_DRUM_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

```
n1:Rhythm Setup #       (0〜3:1〜4)
n2:Voice Reserve        (0〜30)
n3:Receive Channel      (1〜17:1〜16,OFF)
n4:Level                (0〜127)
n5:Level Boost Sw       (0〜1:OFF/ON)
n6:Rx Volume            (0〜1:OFF/ON)
n7:Rx Hold              (0〜1:OFF/ON)
```

(U220マニュアルp63,p149参照)

例

```
[U220_DRUM_SETUP $10 {0,5,10,127,0,1,0}]
```

MIDI U220ドラムセットノートパラメータの設定

[U220_DRUM_INST n,id {n1,n2,…,n20}]

・n:35-99
・id:0-127

・n1〜n20:0-127
U220のテンポラリドラムセットの各ノートにおけるパラメータを設定する。
nは設定先のノート番号で、設定範囲は35≦n≦99。
idはU220のデバイスIDで省略可能、省略時は$10。
パラメータ(n1〜n20)は20個以内任意の個数設定可能。同機能の共通コマンド.U220_DRUM_INSTとは異なり複数行にパラメータを分けて書くことはできない。

n1:Tone Media	(0〜31:I,1〜31)
n2:Tone Number	(1〜128)
n3:Source Key	(0〜127:C−1〜G9)
n4:Mute Inst	(34〜98:OFF,B1〜D7)
n5:Inst Level	(0〜31)
n6:Velocity Sens	(0〜15)
n7:Env Mode	(0〜1:Sustain,No Sustain)
n8:Env Attack Rate	(1〜15:−7〜+7)
n9:Env Decay Rate	(1〜15:−7〜+7)
n10:Env Release Rate	(1〜15:−7〜+7)
n11:Pitch Shift Coarse	(0〜27:-36,-24,-12〜+12)
n12:Pitch Shift Fine	(14〜114:−50〜+50)
n13:Channel After Sens	(0〜27:-36,-24,-12〜+12)
n14:Poly After Sens	(0〜27:-36,-24,-12〜+12)
n15:Random	(0〜15)
n16:Auto Bend Depth	(0〜27:-36,-24,-12〜+12)
n17:Auto Bend Rate	(0〜15)
n18:Detune Depth	(0〜15)
n19:Output Assign	(0〜3:Dry,Rev,Cho,Dir1)
n20:Pan	(0〜15:L7〜M〜R7)

(U220マニュアルp72,p149参照)

例

[U220_DRUM_INST 42,$10 {0,1,24,34,31,15,1,8,8,8,14,64,14,14,
0,14,0,0,0,8}]

(本来はコマンドライン途中で改行することはできないがレイアウトの都合上改行している)

MIDI　U220ディスプレイ文字列表示
[U220_PRINT id　str]

・id:0-127
・str:文字列
U220のディスプレイに文字列を表示する。具体的にはU220のテンポラリパッチの名前を設定することで行う。
idはU220のデバイスIDで省略可能、省略時は$10。
strは表示文字列を表し、"〜"で括り、12文字以内で指定する。

例

[U220_PRINT $10 "APPLE TREE"]

■ M1/M1EX/M1R/M1REX専用命令

MIDI　M1受信チャンネル設定
[M1_SETUP {ch1,ch2,…,ch8}]
[M1_MIDI_CH {ch1,ch2,…,ch8}]

・ch1〜ch8:1-16
M1のSEQ0の各パートの受信チャンネルを設定する。
パラメータ(ch1〜ch8)は必ず8個指定しなければならない。設定範囲は1≦chi≦17で1〜16はMIDIチャンネルの1〜16を表し、17は「そのパートは受信しない」を意味する設定となる。

n1:パート1の受信MIDIチャンネル
n2:パート1の受信MIDIチャンネル
:
:
n8:パート1の受信MIDIチャンネル

例

[M1_SETUP {1,2,3,4,5,6,17,10}]

MIDI　M1パートパラメータの設定
[M1_PART_SETUP {n1,n2,…,n40}]

・n1〜n40:
M1のSEQ0の各パートのパラメータを設定する。
パラメータ(n1〜n40)は必ず40個(5×8パート分)設定しなければならない。また同機能の共通コマンドM1_PART_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

PART 1

n1:PROGRAM NUMBER	(0〜199:I00〜C99)
n2:OUTPUT LEVEL	(0〜99)
n3:KEY TRANSPOSE	($F4〜$0C:−12〜12)
n4:DETUNE	($CE〜$32:−50〜+50)
n5:PAN	(0〜13:10:0〜0:10,C,C+D,D)

PART 2〜8(n6〜n40)以下同様の書式構成
(M1マニュアルp126参照)

例

[M1_PART_SETUP {0,99,0,0,5,0,99,0,0,5,0,99,0,0,5,0,99,0,0,5
0,99,0,0,5,0,99,0,0,5,0,99,0,0,5,0,99,0,0,5}]

(本来はコマンドライン途中で改行することはできないがレイアウトの都合上改行している)

MIDI　M1エフェクトパラメータの設定
[M1_EFFECT_SETUP {n1,n2,…,n25}]

・n1〜n25:
M1のSEQ0のエフェクトパラメータの設定を行う。
パラメータ(n1〜n25)は必ず25個指定しなければならない。
同機能の共通コマンドM1_EFFECT_SETUPとは異なり複数行にパラメータを分けて書くことはできない。

n1:Effect 1 Pattern No.	(0〜33:1〜32,Thru)
n2:Effect 2 Pattern No.	(0〜33:1〜32,Thru)
n3:Effect 1 L-Ch E.Balnc	(0〜100)
n4:Effect 1 R-Ch E.Balnc	(0〜100)
n5:Effect 2 L-Ch E.Balnc	(0〜100)
n6:Effect 2 R-Ch E.Balnc	(0〜100)
n7:Output 3 Pan	(0〜101:OFF,100:0〜0:100)
n8:Output 4 Pan	(0〜101:OFF,100:0〜0:100)
n9:Effect I/O	(BIT0:Effect 1 L-Ch 0=OFF/1=ON BIT1:Effect 1 R-Ch 0=OFF/1=ON BIT2:Effect 2 L-Ch 0=OFF/1=ON BIT3:Effect 2 R-Ch 0=OFF/1=ON) BIT4:Effect 2 0=PARALLEL/1=SERIAL)
n10〜n17:Effect 1 Parameter	
n18〜n25:Effect 2 Parameter	

Structure of Effect Parameter
(ofs=10 or 18)

1〜3:Hall (4,5:Room,6:Live Stage)

n(ofs+0):Reverb Time	(0〜97:0.2〜9.9),(0〜48:0.2〜5.0)
n(ofs+1):Dummy	0
n(ofs+2):High Damp	(0〜99)
n(ofs+3):Pre Delay	(0〜200)
n(ofs+4):E/R Level	(0〜99)
n(ofs+5):Dummy	0
n(ofs+6):EQ High	($F4〜$0C:−12〜+12)
n(ofs+7):EQ Low	($F4〜$0C:−12〜+12)

7〜9:Early Reflection 1,2,3

n(ofs+0):E/R Time	(0〜70:100〜800)
n(ofs+1):Pre Delay	(0〜200)
n(ofs+2):Dummy	0
:	:
n(ofs+5):Dummy	0
n(ofs+6):EQ High	($F4〜$0C:−12〜+12)
n(ofs+7):EQ Low	($F4〜$0C:−12〜+12)

10:Stereo Delayt,11:Cross Delay

n(ofs+0):Delay Time L (L)	(0〜500)
n(ofs+1):Delay Time L (H)	
n(ofs+2):Feedback	($9D〜$63:−99〜99)
n(ofs+3):High Damp	(0〜99)
n(ofs+4):Delay Time R (L)	(0〜500)
n(ofs+5):Delay Time R (H)	
n(ofs+6):EQ High	($F4〜$0C:−12〜+12)
n(ofs+7):EQ Low	($F4〜$0C:−12〜+12)

12〜13:Stereo Chorus 1〜2 (14〜15:Flanger)

n(ofs+0):Depth (0～99)
n(ofs+1):Speed (0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status (BIT0:Waveform =0:Sin,=1:Tri
BIT1:Phase =0:0.0°,=1:180°
BIT2:Wave Shape =0:Normal
=1:for Flanger)
n(ofs+3):Feedback ($9D～$63:－99～＋99)
n(ofs+4):Delay Time (0～200),(0～50)
n(ofs+5):Dummy 0
n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

16～17:Phase Shifter 1～2
n(ofs+0):Depth (0～99)
n(ofs+1):Speed (0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status (BIT0:Waveform =0:Sin,=1:Tri
BIT1:Phase =0:0.0°,=1:180°
BIT2:Wave Shape =0:Normal
=1:for Flanger)
n(ofs+3):Feedback ($9D～$63:－99～＋99)
n(ofs+4):Manual (0～99)
n(ofs+5):Dummy 0
n(ofs+6):Dummy 0
n(ofs+7):Dummy 0

18～19:Stereo Toremolo 1～2
n(ofs+0):Depth (0～99)
n(ofs+1):Speed (0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
n(ofs+2):LFO Status (BIT0:Waveform =0:Sin,=1:Tri
BIT1:Phase =0:0.0°,=1:180°
BIT2:Wave Shape =0:Normal
=1:for Flanger)
n(ofs+3):Shape ($9D～$63:－99～＋99)
n(ofs+4):Dummy 0
n(ofs+5):Dummy 0
n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

20:Equalizer
n(ofs+0):Dummy 0
: :
n(ofs+3):Dummy 0
n(ofs+4):Low fc (0～2:0.25k,0.50k,1.00k)
n(ofs+5):High fc (0～2:1k,2k,4k)
n(ofs+6):High Gain ($F4～$0C:－12～＋12)
n(ofs+7):Low Gain ($F4～$0C:－12～＋12)

21:Overdrive
n(ofs+0):Dummy 0
n(ofs+1):Dummy 0
n(ofs+2):Drive (0～99)
n(ofs+3):Level (0～99)
n(ofs+4):Dummy 0
n(ofs+5):Dummy 0
n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

22:Distortion
n(ofs+0):Dummy 0
n(ofs+1):Dummy 0
n(ofs+2):Distortion (0～99)
n(ofs+3):Level (0～99)
n(ofs+4):Dummy 0
n(ofs+5):Dummy 0
n(ofs+6):Dummy 0
n(ofs+7):EQ Low Gain ($F4～$0C:－12～＋12)

23:Exciter
n(ofs+0):Blend ($9D～$63:－99～＋99)
n(ofs+1):Emphatic Point (0～9:1～10)
n(ofs+2):Dummy 0
: :
n(ofs+5):Dummy 0
n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

24:Synphonic Ensamble
n(ofs+0):Depth (0～99)
n(ofs+1):Dummy 0
: :
n(ofs+5):Dummy 0
n(ofs+6):EQ High ($F4～$0C:－12～＋12)
n(ofs+7):EQ Low ($F4～$0C:－12～＋12)

25:Roteffect Speaker
n(ofs+0):Depth (0～99)
n(ofs+1):Dummy 0
n(ofs+2):Speed Rate ($F6～$0A:－10～＋10)
n(ofs+3):Dummy 0
: :
n(ofs+7):Dummy 0

26:Delay/Hall
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Reverb Time (0～97:0.2～9.9)
n(ofs+5):Dummy 0
n(ofs+6):High Damp (0～99)
n(ofs+7):Pre Delay (0～150)

27:Delay/Room
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Reverb Time (0～97:0.2～9.9)
n(ofs+5):Dummy 0
n(ofs+6):High Damp (0～99)
n(ofs+7):Pre Delay (0～150)

28:Delay/Early Reflection
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):E/R Time (0～30:100～400)
n(ofs+5):Pre Delay (0～150)
n(ofs+6):Dummy 0
n(ofs+7):Dummy 0

29:Delay/Chorus
n(ofs+0):Delay Time L (L) (0～500)
n(ofs+1):Delay Time L (H)
n(ofs+2):Feedback L ($9D～$63:－99～99)
n(ofs+3):High Damp L (0～99)
n(ofs+4):Delay Time R (L) (0～500)
n(ofs+5):Delay Time R (H)
n(ofs+6):Feedback R ($9D～$63:－99～99)
n(ofs+7):High Damp R (0～99)

30:Delay/Chorus (31:Delay Flanger)
n(ofs+0):Delay Time (L) (0～500)

n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
n(ofs+6):LFO Status (BIT0:Waveform =0:Sin,=1:Tri
BIT1:Phase =0:0.0°,=1:180°
BIT2:Wave Shape =0:Normal
=1:for Flanger)
n(ofs+7):Feedback 0,($9D～$63:－99～99)

32:Delay/Phaser
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step)
n(ofs+6):Feedback ($9D～$63:－99～99)
n(ofs+7):Dummy 0

33:Delay/Tremolo
n(ofs+0):Delay Time (L) (0～500)
n(ofs+1):Delay Time (H)
n(ofs+2):Feedback ($9D～$63:－99～99)
n(ofs+3):High Damp (0～99)
n(ofs+4):Depth (0～99)
n(ofs+5):Speed (0～99:0.03～3.00 0.03step)
n(ofs+6):Dummy 0
n(ofs+7):Shape ($9D～$63:－99～99)
(M1マニュアルp127参照)

例

[M1_EFFECT_SETUP {1,3,50,50,50,50,50,50,%11111,
50,0,50,100,50,0,0,0,50,0,50,100,50,0,0,0}]

(本来はコマンドライン途中で改行することはできないがレイアウトの都合上改行している)

MIDI　M1ディスプレイ文字列表示
[M1_PRINT str]

・str:文字列

M1のディスプレイに文字列を表示する。具体的にはM1のSEQ0のソングネームの設定をすることで行う。

strは表示文字列で、"～"で括り、10文字以内で指定する。

例

[M1_PRINT "MY WAY"]

MIDI　M1へ設定を送信する
[SEND_TO_M1 id]

・id:0-127

[M1_SETUP]
[M1_PART_SEUP]
[M1_EFFECT_SETUP]
[M1_PRINT]

で設定したパラメータ群をM1へ送信する。

idはM1のデバイスIDで省略可能、省略時は$30。

なお、M1のデバイスIDは$30＋グローバルチャンネル(0～$f)で求められる(グローバルチャンネルはGLOBALモードのF5－1で設定できる)。

必ず、

[M1_SETUP]
[M1_PART_SEUP]
[M1_EFFECT_SETUP]
[M1_PRINT]

を設定してからこの命令を実行すること。[M1_EFFECT_SETUP]のみ省略が可能。省略時はZ-MUSICが持つデフォルト設定がM1へ送信され設定されることになる。

[M1_SETUP]
[M1_PART_SEUP]
[M1_EFFECT_SETUP]
[M1_PRINT]

の間にM1関係以外のコマンドがあると正常なデータが送信されないので注意すること。

例

[SEND_TO_M1 $30]

ver.2とver.3でのMMLの違い

Z-MUSIC ver.2.0以前とZ-MUSIC ver.3.0以降とでは機能拡張にともない、以下の２点において動作が微妙に異なります

1.波形メモリ

振幅を与えることができるようになった関係で、波形メモリを使用した曲では演奏した感じが異なる場合がある。

2.絶対音長１

絶対音長1がV2以前ではOPMDRV.Xとの互換を保つために強制的に「タイ/スラー」指定としていたが汎用性を考えこれを取り止めた。

ver.2.0の演奏データであってもほとんどのものはそのまま演奏することができます。しかし、微妙に演奏のニュアンスが異なっていた場合は、ZMSC3.X に'-2'オプションをつけて常駐し直してから再び演奏させてみてください。'-2'オプションはZMSC3.Xをver2.0相当の動作をさせるスイッチです。当然ながら'-2'オプションを設定してしまうと今度は、ver3.0以降用の演奏データが正常に演奏されなくなってしまいます。

MEASURE 6

ZPCNV3.RとZPLK3.R

ここではZ-MUSIC ver.3.0での(AD)PCMデータの取り扱いと、
支援ツール「ZPCNV3.R」「ZPLK3.R」について解説します。

6．1．　はじめに

Z-MUSIC ver.3.0には起動時(AD)PCMデータはひとつもプリセットされていない。そのため(AD)PCM音を鳴らすためには鳴らしたい(AD)PCMファイルを登録することが必要になる。

Z-MUSIC ver.3.0では(AD)PCMの登録形式は「トーン形式」「音色方式」の２つがある。「トーン形式」はver.2.0以前からある形式で、鍵盤ひとつひとつに任意の音声データを割り当てていくもの。「音色方式」はver.3.0になって新設された形式で、MPCM.Xを常駐時にのみ有効になる。これは音声データをひとつ登録しておけば、これを自在に音程変換して発音することができる方式である。トーン形式(またはver.2.0以前)ではたとえばピアノの音でド・レ・ミを演奏するにはド・レ・ミの３つの音程の音を用意してZ-MUSICに登録しなければならなかったが、「音色方式」ではドの音を用意して、「この音はド」と指定して登録すれば任意の音程の音が発音できるようになる。

一方、Z-MUSIC ver.3.0に(AD)PCMデータを渡す手段は３通りあり、ひとつは(AD)PCM登録コマンドをミュージックプログラムに直接記述していく方法。２つめは(AD)PCM登録ZMSコマンドを記述したファイルをエディタ(ED.Xなど)で作成しこのファイルを渡す方法である。３つ目はそのファイルをZPCNV3.RでZPDファイルにコンバートして、これをZ-MUSIC ver.3.0に渡す方法である。

6．2．　(AD)PCMコンフィギュレーションファイル

(AD)PCM登録ZMSコマンドを箇条書きにしたファイルを、特に「(AD)PCMコンフィギュレーションファイル」と呼び、これをZ-MUSIC ver.3.0へ受け渡すことで(AD)PCMデータを登録することができる。奨励ファイル拡張子は'.CNF'で、このファイルを特にCNFファイルと呼ぶ。

・コンフィギュレーションファイルの例

```
.o1b     conbd.pcm,v50
.o2e     nnsdd.pcm,v70
.o3g+    cowbell1.pcm,v60
.o3a     Acrash.pcm
.o4g     agohi.pcm
.o5a     triangle.pcm

.16bitPCM_TIMBRE 0,61,A#3
{
    FHRNA#3at.P16
    .CONNECT FHRNA#3LP.P16
    .v50
}
.16bitPCM_TIMBRE 0,65,A#5
{
    asxa#5at.P16
    .CONNECT asxa#5lp.P16
    .v50
}
.16bitPCM_TIMBRE 0,17,b2
{
    porgb2at.p16
    .CONNECT porgb2lp.p16
    .V40
}
.ADPCM_TIMBRE 0,39,c2     {EBASS303.PCM,v60}
.ADPCM_TIMBRE 0,62,e4     {BRASS_E4.PCM}
.ADPCM_TIMBRE 0,49,c5     {strsbc.pcm}
.ADPCM_TIMBRE 0,13,d3     {mrmba_d3.pcm,v80}
.ADPCM_TIMBRE 0,5,c6      {prel_c6.pcm,v50}
.ADPCM_TIMBRE 0,59,c2     {tuba.pcm,v30}
.ADPCM_TIMBRE 0,73,b4     {ahbr.pcm,v50}
.ADPCM_TIMBRE 0,23,c4     {harmonica.pcm,v60}
```

6．3．(AD)PCMコンフィギュレーションファイルの登録方法

MUSICZ3.FNCを組み込んだX-BASIC上から指定したい場合には関数、

　zm_pcm_read()

を使う(MEASURE3参照)。

ZMS中に指定したい場合はZMSコマンド、

　.ADPCM_LIST filename

を用いる(MEASURE4参照)。

また、演奏データのように、

　A>copy (AD)PCMコンフィギュレーションファイルネーム ZMS

としたり、

　A>zp (AD)PCMコンフィギュレーションファイルネーム

としたりして、演奏データを演奏するような操作でZ-MUSIC ver.3.0へ登録することも可能(MEASURE7参照)。こちらの場合は曲がたとえ演奏中であってもリアルタイムに(AD)PCMセットを変更することができる。

6．4．　ZPCNV3.RとZPD

6．4．1．ZPDの作成方法

ZPCNV3.Rは、6.1.で解説したコンフィギュレーションファイルに従って必要な(AD)PCMデータをひとつのファイルにまとめるプログラムである。ひとまとめにする際に音程音量変換、合成作業といった各種加工処理もしてしまうため、実際の(AD)PCMデータの登録処理にかかる時間は極めてゼロに近い((AD)PCMコンフィギュレーションファイルをZ-MUSIC ver.3.0へ渡す方式では１個１個のデータを読み込み、それから加工処理を行ったりするため、大量の(AD)PCMデータを登録する場合には処理が終了するまで長時間待たされることがある)。

ZPCNV3.Rは、加工処理ルーチンやその際に使用するワークは独自に備えているため実行にはZ-MUSIC ver.3.0の常駐は必要ない。

Z-MUSIC ver.3.0では(AD)PCMコンフィギュレーションファイルをもとに、ひとまとめにしたこのデータを(AD)PCMブロックデータ、「ZPD」と呼ぶ。

たとえばfilename1.CNFで表されるコンフィギュレーションファイルをZPDに変換したい場合は、

　A>ZPCNV3.R filename1 filename2

とする。filename1を指定する際、拡張子を省略すると'.CNF'が自動添付される。filename2は作成するZPDのファイル名でこちらは省略することが可能。省略時はfilename1に拡張子'.ZPD'をつけたものがfilename2として与えられる。

6．4．2．ZPCNV3.Rのコマンドオプション

■日本語メッセージの表示

●日本語メッセージの表示
－J
ZPCNV3.Rが出力するメッセージを日本語にする。本コマンドオプションは必ずコマンドラインの先頭に記述しなければ有効とならない。
例
```
A>ZPCNV3 -j STD_SET.CNF STD_SET.ZPD
```

6．4．3．環境変数について

環境変数'zpcnv3_opt'に通常よく設定するコマンドオプションをあらかじめ設定しておくことができる。
たとえば、常に日本語メッセージでメッセージ出力を行いたい場合はAUTOEXEC.BATなどに、
```
SET zpcnv3_opt=-J
```
と設定しておく。そうすれば、
```
A>ZPCNV3 STD_SET.CNF STD_SET.ZPD
```
と実行しただけでも -J が追加され、
```
A>ZPCNV3 -J STD_SET.CNF STD_SET.ZPD
```
として実行される。

6．5．ZPDのZ-MUSIC ver.3.0への登録法

MUSICZ3.FNCを組み込んだX-BASIC上から指定したい場合には関数、
```
zm_register_zpd()
```
を使う(MEASURE3参照)。
ZMS中に指定したい場合はZMSコマンド、
```
.ADPCM_BLOCK_DATA  filename
.ZPD  filename
```
を用いる(MEASURE4参照)。
また、通常の演奏データのように、
```
A>copy filename.ZPD  ZMS
```
としたり、
```
A>zp filename.ZPD
```
としたりして、通常の演奏データを演奏させるような手法でもZ-MUSIC ver.3.0へ登録することができる(MEASURE7参照)。こういった手段で登録する場合は、曲がたとえ演奏中であってもリアルタイムにZPDを変更することができる。
また、ZMSC3.Xの常駐スイッチの'-Z'を用いて、
```
A>ZMSC3 -Zdrum_set.zpd     (拡張子の省略は可能)
```
のようにZMSC3.X常駐時にZPDを登録することもできる。

6．6．PCMファイルリンカ「ＺＰＬＫ３．Ｒ」

ZPLK3.RはZ-MUSIC ver.3.0から追加された(AD)PCMデータの加工/リンクツールである。以下にその特徴を示す。

6．6．1．その機能と特長

・最大32個の複数のファイルをつなげて１個のファイルにして出力できる。ひとつのファイルにつき最大65535回の反復指定が可能。反復パターンは4種類。
・単一(AD)PCMファイルをいくつもつなげて１個の大きな(AD)PCMファイルにすることが可能。
・入力ファイルとして16ビットPCMファイル、AD PCMファイルを与えることができる。混在も可能。
・リンク作業の前に、各入力ファイルのデータ形式コンバートを行うことができるので16ビットPCMに(AD)PCMファイルをリンクさせたり、その逆も行える。
・出力ファイルはその出力直前に16ビットPCMデータコンバート、(AD)PCMデータコンバート、逆転再生、音量変換、周波数変換、ポルタメント、エンベロープ変更など多彩な加工処理を行うことができる。
・インパルスデータを与えることにより、「畳み込み演算」を行うことができる。したがって、任意の音声データに対してホールの残響効果やボコーダ処理などの本格的なエフェクト処理をすることができる。

6．6．2．ZPLK3.Rのコマンドオプション

ZPLK3.R <[オプション1] 入力ファイル１> [[オプション2～32] 入力ファイル２～32] <出力ファイル名>
入力ファイルネームは最低１個は必要となる。書き込みファイル名は'PCM'に設定するとファイルを書き出さず、音声として出力できる(ZMSC3.XやPCMDRV.SYS組み込み時)。ファイルネームの指定は拡張子やパスの省略をせずきちんと書くこと。ファイルがカレントより見つからない場合は環境変数'zmusic'に書かれたパスに従ってファイルを検索する。

●コマンドオプション

-A　出力ファイルを出力する前にそのデータを16ビットPCMデータと見なして(AD)PCMデータへ変換する。

-Bs,d,p,s　出力ファイルを出力する前にそのデータを16ビットPCMデータと見なしてポルタメントを行う。(連続的に滑らかに周波数変更を行う。)
s　開始時の周波数をs[Hz]とみなす。ただしsは1～65535。
d　終了時の周波数をd[Hz]とする。ただしdは1～65535。
p　出力ファイルのpバイト目から周波数変換を行う。省略時は、p=0。
s　変換するサイズをsバイトとする。省略時はpで指定された位置から後ろ全部を変換対象領域とする。pも省略した場合は出力全域に対して変換処理を施す。

-Cp,s　出力ファイルを出力する前にそのデータの任意の一部分を摘出しそれを出力する。
p　出力ファイルのpバイト目から切り出す。省略時はp=0。
s　切り出すサイズをsバイトとする。省略時はpで指定された位置から後ろ全部を切り出す。pも省略した場合はエラーとなる。

-Fs,l,m　出力ファイルを出力する前にそのデータを16ビットPCMデータと見なしてエンベロープの形状を変化させる。
s　出力ファイルのsバイト目からエンベロープ形状を変更する。省略時はs=0。
l　エンベロープ形状初期音量(m=0時)/最終音量(m=1時)を0～127の128段階で設定する。省略時はl=0。
m　エンベロープの変更パターンを設定する。省略時はm=1
0 フェードインタイプ
1 フェードアウトタイプ

-Ifilename　インパルスデータのファイルネームを与えると出力ファイルを作成する前にそのデータとの畳み込み演算を行う。ただし、与えられたファイルネームの拡張子を'.P16'とした場合はそのインパルスデータを16ビットPCMデータ、'.PCM'とした場合は(AD)PCMデータとして見なす。

-P　出力ファイルを作成する前にそのデータを(AD)PCMデータと見なして16ビットPCMデータへ変換する。

-R　出力ファイルを作成する前にそのデータを16ビットPCMデータと見なして逆転させる。

-Vn　出力ファイルを作成する前にそのデータの音量をパーセント単位で指定する。nの範囲は0～300。スイッチ無指定や値省略時は100とする。

-Ti,o,p,s　出力ファイルを出力する前にそのデータを16ビットPCMデータと見なして周波数の変更を行う。
i　元の周波数をs[Hz]と見なす。ただしsは1～65535。
o　変換後の周波数をd[Hz]とする。ただしdは1～65535。
p　出力ファイルのpバイト目から周波数変換を行う。省略時は、p=0。
s　変換するサイズをsバイトとする。省略時はpで指定された位置から後ろ全部を変換対象領域とする。pも省略した場合は出力全域に対して変換処理を施す。

-Xl,r,t　リンク制御
l　ループタイプの設定。省略時 l=0
0 そのまま

1 正順、逆順の順にループさせる
2 逆順、正順の順にループさせる
3.入力ファイルの逆順をループさせる

r　反復回数の設定。省略時 r=1。範囲1～65535

t　リンク作業の前に入力ファイルに対して加工を行う。
(省略時はt=0)
0 なにも行わない
1 入力ファイルを16ビットPCMデータと見なして(AD)PCMデータへ変換する
2 入力ファイルを(AD)PCMデータと見なして16ビットPCMデータへ変換する

●使用例

```
A>zplk a.p16 -x0,10 b.p16 -a c.pcm
```

a.p16 というファイルの後ろにb.p16を10回追加して、AD PCMへコンバートし c.pcmというファイル名でセーブする。

```
A>zplk -x0,5 a.p16 -x,,2 b.pcm -v80 c.p16
```

a.p16 というファイルを５回追加し、これにb.pcmを16ビットPCMへ変換したものを接続して、それを音量を80%に変換しc.p16というファイル名でセーブする。

【注意】

複数の変換を指定した場合は以下の手順で各変換を行う。

音量変更周波数変換
↓
ポルタメント
↓
切り出し
↓
逆転
↓
フェードイン/フェードアウト
↓
インパルス演算
↓
16ビットPCM/(AD)PCM変換

●パラメータの特殊設定

周波数変換(-Tスイッチ)、ポルタメント変換(-Bスイッチ)において、２つの周波数パラメータの設定をする代わりにZ-MUSIC ver.3.0システムのZMSコマンド表記に近い半音ステップの変換パラメータを設定することもできる。

この指定を行うには、'-T','-B'のオプションコマンドの後ろにさらにサブスイッチ'p'(大文字可)を書いてから変換値を設定する。

たとえば、ある音を半音下げたいときには、

-Tp-1

また、半音上げたいときは、

-Tp+1

とする。

また、ポルタメントのときも同様で、

-Bp-10
-Bp+6

といった記述ができる。サブスイッチ'p'の値の範囲は-144～+144(ただし0は除く)。

●参考　国際基準イ＝a1＝440Hzに基づく12平均率音階周波数表
(試験に出るX1より転載)

	00	01	02	03	04	05	06	07	08
C	16.352	32.703	65.406	130.81	261.63	523.25	1046.5	2093.0	4186.0
C＃	17.324	34.648	69.296	138.59	277.18	554.37	1108.7	2217.5	4434.9
D	18.354	36.708	73.416	146.83	293.66	587.33	1174.7	2349.3	4698.6
D＃	19.445	38.891	77.782	155.56	311.13	622.25	1244.5	2489.0	4978.0
E	20.602	41.203	82.407	164.81	329.63	659.26	1318.5	2637.0	5274.0
F	21.827	43.654	87.307	174.61	349.23	698.46	1396.9	2793.8	5587.7
F＃	23.125	46.249	92.499	185.00	369.99	739.99	1480.0	2960.0	5919.9
G	24.500	48.999	97.999	196.00	392.00	783.99	1568.0	3136.0	6271.9
G＃	25.957	51.913	103.83	207.65	415.30	830.61	1661.2	3322.4	6644.9
A	27.500	55.000	110.00	220.00	440.00	880.00	1760.0	3520.0	7040.0
A＃	29.135	58.270	116.54	233.08	466.16	932.33	1864.7	3729.3	7458.6
B	30.868	61.735	123.47	246.96	493.88	987.77	1975.5	3951.1	7902.1

MEASURE 7

ZP3.R

ここではZ-MUSICの演奏データプレイヤー「ZP3.R」について解説します。

7.1. ZP3.Rについて

ZMSC3.Xが常駐しているとき、ZMDを演奏するには、

A>COPY filename.ZMD ZMS

のように、演奏データZMDをシステムファイル名'ZMS'に出力することで行う。

コンパイラZMC.Xを常駐させればZMSファイルであっても、同様な手法で、

A>COPY filename.ZMS ZMS

のようにして演奏させせることができる。

ZP3.Rは以上のような操作を、より簡単に行えるように開発されたZ-MUSIC用演奏データ専用再生ツールである。

また、ZP3.Rは演奏制御機能の他に「ジュークボックス機能」や「キーボード操作によるリアルタイム演奏制御機能」などの機能が備わっている。

7.2. ZP3.Rのコマンドオプション

【注意】

ZP3.Rの英字指定によるコマンドオプションの中には大文字と小文字とで異なった機能が働くように設計されているものもあるので注意。大文字のほうしか解説がないコマンドオプションに関しては大文字、小文字どちらを使用しても機能に相違はない

■演奏制御

●演奏開始

－P 演奏データファイルネーム,セットアップファイル1,…,セットアップファイル4

演奏データをディスクから読み込んで演奏する。演奏データのファイル名の後ろに','で区切って最高4つまでのセットアップファイルを演奏前に読み込み、実行が可能。セットアップファイルとは、たとえば、(AD)PCMデータのコンフィギュレーションファイル(CNF)、ZPDファイル、スタンダードMIDIファイル(MID)などのこと(セットアップファイルのスタンダードMIDIファイルは拡張子'.MID'のときのみ有効)。

セットアップファイルの拡張子は省略できないが演奏データのほうは拡張子を省略でき、その場合は'.ZMD'→'.ZMS'→'.OPM'の順で自動添付してファイルを検索する(見つからなかった場合はエラーとなる)。

例

A>ZP3 -p music.zms, drum.cnf, u220_seup.mid
A>ZP3 -p etude

スイッチ'－P'は省略することもでき、上の例は、

A>ZP3 music.zms, drum.cnf, u220_seup.mid
A>ZP3 etude

とすることもできる。

さらに、演奏データを省略して、ZPDの登録やMIDファイルの転送、(AD)PCMコンフィギュレーションファイルの実行といったセットアップファイルのみの指定も可能である(拡張子の省略はできない)。ZPDの登録と(AD)PCMコンフィギュレーションファイルの実行は曲が演奏中であっても可能なので、演奏中の曲の(AD)PCMのセットをリアルタイムに変更することができる。

例

A>ZP3 list.cnf
((AD)PCMコンフィギュレーションファイルlist.cnfを実行する)
A>ZP3 drum_set.zpd　(ZPDファイルdrum_set.zpdを登録する)
A>ZP3 M1preset.mid　(MIDファイルM1preset.midを楽器へ転送する)

●演奏開始

－Pn1,n2,n3,…ni

演奏を開始する。コマンドスイッチの後ろにトラック番号(1≦ni≦65535)を書くとそのトラックのみ演奏を開始する。トラック番号は、横につなげていくつでも書くことができる。また、トラック番号を書かずにコマンドスイッチのみを指定した場合は演奏可能なすべてのトラックの演奏を開始する。

例

A>ZP3 -p　(全トラック演奏開始)
A>ZP3 -p1,2,10　(トラック1, 2, 10の演奏を開始する)
(MEASURE3 zm_play(), MEASURE4 .PLAY参照)

●演奏停止

－Sn1,n2,n3,…ni

演奏の停止を行う。コマンドスイッチの後ろにトラック番号(1≦ni≦65535)を書くとそのトラックに対して機能する。トラック番号は横につなげていくつでも書くことができる。

また、トラック番号を書かずにコマンドスイッチのみを指定した場合は演奏中の全トラックに対して機能する。

例

A>ZP3 -s　(全トラック演奏停止)
A>ZP3 -s1,2,10　(トラック1,2,10の演奏を停止する)
(MEASURE3 zm_stop() MEASURE4 .STOP参照)

●演奏再開

－Cn1,n2,n3,…ni

演奏の再開を行う。コマンドスイッチの後ろにトラック番号(1≦ni≦65535)を書くとそのトラックに対して機能する。トラック番号は横につなげていくつでも書くことができる。

また、トラック番号を書かずにコマンドスイッチのみを指定した場合は演奏中の全トラックに対して機能する。

例

A>ZP3 -c　(全トラック演奏再開)
A>ZP3 -c1,2,10　(トラック1,2,10の演奏を再開する)
(MEASURE3 zm_cont(), MEASURE4 .CONTINUE参照)

■マスク制御

●トラックマスク解除

－en1,n2,n3,…,ni

指定トラックのマスクを解除する。コマンドスイッチの後ろにトラック番号(1≦ni≦65535)を書くとそのトラックに対して機能する。トラック番号をすべて省略した場合は、マスクをすべて解除(全トラック演奏可能に)する。

実行はリアルタイムに行うことができ、演奏中の曲の任意のトラックのマスク解除を行うことができる。

例

A>ZP3 -e1,2,3
(トラック1～3のマスクを解除、トラック1～3以外をマスクする)
A>ZP3 -e　(全トラックのマスクを解除する)

(MEASURE3 zm_mask_tracks(), MEASURE4 .TRACK_MASK参照)

●チャンネルマスク解除
－En1,n2,n3,…,ni

指定チャンネルのマスクを解除する。チャンネルの指定は下の表で示される文字列を用いて行う。

文字列	対象デバイス
FM1～8	FM音源チャンネル1～8
ADPCM1～16	(AD)PCMチャンネル1～16
PCM1～16	(AD)PCMチャンネル1～16
MIDI1～16	カレントMIDIチャンネル1～16
MIDI1-1～1-16	MIDIインタフェイス1のMIDIチャンネル1～16
MIDI2-1～2-16	MIDIインタフェイス2のMIDIチャンネル1～16
MIDI3-1～3-16	MIDIインタフェイス3のMIDIチャンネル1～16
MIDI4-1～4-16	MIDIインタフェイス4のMIDIチャンネル1～16

チャンネル指定をすべて省略した場合は、マスクをすべて解除(全チャンネル演奏可能に)する。実行はリアルタイムに行うことができ、演奏中の曲の任意のチャンネルのマスク解除を行うことができる。

例

A>ZP3 -EFM1,MIDI2-1
(FM音源ﾁｬﾝﾈﾙ1とMIDIインタフェイス2のﾁｬﾝﾈﾙ1以外をマスクする)
A>ZP3 -E (全チャンネルのマスクを解除する)
(MEASURE3 zm_mask_channels()参照)

●トラックマスク設定
－mn1,n2,n3,…,ni

指定トラックの演奏をマスクする。それ以外のトラックは通常演奏を続行する。コマンドスイッチの後ろにトラック番号(1≦ni≦65535)を書くとそのトラックに対して機能する。トラック番号をすべて省略した場合は、マスクをすべて解除(全チャンネル演奏可能に)する。実行はリアルタイムに行うことができ、演奏中の曲の任意のトラックをマスクすることができる。

例

A>ZP3 -m1,2,3 (トラック1～3をマスクする)
A>ZP3 -m (全トラックのマスクを解除する)
(MEASURE3 zm_mask_tracks(), MEASURE4 .TRACK_MASK参照)

●チャンネルマスク設定
－Mn1,n2,n3,…,ni

指定チャンネルの演奏をマスクする。それ以外のチャンネルは通常演奏を続行する。チャンネルの指定は下の表で示される文字列を用いて行う。

文字列	対象デバイス
FM1～8	FM音源チャンネル1～8
ADPCM1～16	(AD)PCMチャンネル1～16
PCM1～16	(AD)PCMチャンネル1～16
MIDI1～16	カレントMIDIチャンネル1～16
MIDI1-1～1-16	MIDIインタフェイス1のMIDIチャンネル1～16
MIDI2-1～2-16	MIDIインタフェイス2のMIDIチャンネル1～16
MIDI3-1～3-16	MIDIインタフェイス3のMIDIチャンネル1～16
MIDI4-1～4-16	MIDIインタフェイス4のMIDIチャンネル1～16

チャンネル指定をすべて省略した場合は、マスクをすべて解除(全チャンネル演奏可能に)する。実行はリアルタイムに行うことができ、演奏中の曲の任意のチャンネルのマスク設定を行うことができる。

例

A>ZP3 -MFM1,ADPCM2,MIDI3
(MEASURE3 zm_mask_channels()参照)

■フェーダー操作

●フェードイン/フェードアウト
－Fn

フェードイン/アウトを行う。-255≦n≦-1のときフェードイン、1≦n≦255のときフェードアウトの指定となり、数値の絶対値が大きいほど音量の増減スピードは速くなる。n=0はフェードイン/アウトの停止の指定となる。

パラメータnを省略した場合はデフォルト値のスピードでフェードアウトを行う。

なお、本オプションで操作されるフェーダーはマスターフェーダー。

例

A>ZP3 -f20 (フェードアウトをフェードアウトスピード20で行う)
A>ZP3 -f-16 (フェードインをフェードインスピード16で行う)
A>ZP3 -f0 (フェードイン/アウトを中断する)
A>ZP3 -f (デフォルトスピードでフェードアウトを行う)
(MEASURE3 zm_master_fader(), MEASURE4 .MASTER_FADER参照)

●トラックフェーダー操作
－Olv,n1,n2,…,ni

指定トラックのフェーダー(出力音量割合)を設定する。

lvは出力音量の割合で、設定範囲は0≦lv≦128。128が最大音量かつ基本状態。lvの値が小さいほど音量も小さくなる。lvによって決定づけられる実際の音量は各楽器タイプや音源によって違うので注意。

niは出力割合lvの設定を行うトラック番号で、設定範囲は1～65535。

lvのみを設定し、トラック番号をすべて省略した場合は、全トラックが出力レベルlvに設定される。

全パラメータを省略した場合は、全トラックの出力割合を基本状態(lv=128)へ戻す。

実行はリアルタイムに行うことができ、演奏中の曲の任意のトラックの出力割合を変更することができる。

例

A>ZP3 -o110,1,2,3
(トラック1, 2, 3の出力割合を110/128に設定する)
A>ZP3 -o96 (全トラックの出力割合を96/128に設定する)
A>ZP3 -o
(全トラックの出力割合を通常状態(128/128)に戻す)
(MEASURE3 zm_master_fader(), MEASURE4 .MASTER_FADER参照)

●チャンネルフェーダー操作
－Olv,n1,n2,…,ni

指定チャンネルのフェーダー(出力音量割合)を設定する。

lvは出力音量の割合で、設定範囲は0≦lv≦128。128が最大音量かつ基本状態。lvの値が小さいほど音量も小さくなる。lvによって決定づけられる実際の音量は各楽器タイプや音源によって違うので注意。

niは出力割合lvの設定を行うチャンネル。チャンネルの指定は下の表で示される文字列を用いて行う。

文字列	対象デバイス
FM1～8	FM音源チャンネル1～8
ADPCM1～16	(AD)PCMチャンネル1～16
PCM1～16	(AD)PCMチャンネル1～16
MIDI1～16	カレントMIDIチャンネル1～16
MIDI1-1～1-16	MIDIインタフェイス1のMIDIチャンネル1～16
MIDI2-1～2-16	MIDIインタフェイス2のMIDIチャンネル1～16
MIDI3-1～3-16	MIDIインタフェイス3のMIDIチャンネル1～16
MIDI4-1～4-16	MIDIインタフェイス4のMIDIチャンネル1～16

lvのみを設定し、チャンネル指定をすべて省略した場合は、全チャンネルが出力レベルlvに設定される。

全パラメータを省略した場合は、全チャンネルの出力割合を基本状態(lv=128)へ戻す。

実行はリアルタイムに行うことができ、演奏中の曲の任意のチャンネルの出力割合を変更することが可能。

例

A>ZP3 -o110,FM1,MIDI1-2,ADPCM3
(FM音源ﾁｬﾝﾈﾙ1, MIDIインタフェイス1のﾁｬﾝﾈﾙ2, AD PCM音源ﾁｬﾝﾈﾙ3の出力割合を110/128に設定する)
A>ZP3 -o96 (全チャンネルの出力割合を96/128に設定する)
A>ZP3 -o (全チャンネルの出力割合を通常状態(128/128)に戻す)

■初期化

●初期化
－I

Z-MUSIC ver.3.0の演奏ワークや各音源を初期化する。曲が演奏中であれば演奏を停止してから処理を行う。

例

A>ZP3 -i
(MEASURE3 zm_init(), MEASURE4 .INITIALIZE参照)

■ステータスの表示

●ステータスの表示
－Q
－Qファイルネーム

'－Q'で、演奏中やその時点でZ-MUSICが演奏中の演奏データのステップタイム合計値などのステータス値を表示する。ZMDやZMSといった演奏データファイル名を後ろにつければ、それを読み込みステータスを表示してから演奏を開始する。なお、ステータスは以下のような書式で表示される。

```
-TRACK1-----------------------------------------  ........(1)
TOTAL STEPTIME:384;6528(00000180;00001980)        ........(2)
TOTAL CHECKSUM:56790(0000DDD6)                    ........(3)
TOTAL MEASURE :2(00000002)                         .......(4)
-MEASURE1-                                        ........(5)
        STEPTIME:768(00000300)                    ........(6)
        CHECKSUM:3164(00000C5C)                   ........(7)
-MEASURE2-
        STEPTIME:0(00000000)
        CHECKSUM:207356588(0C5C02AC)
-TRACK2-----------------------------------------
        :       トラック1と同様の書式で
        :       トラック数分続く
-TRACK3-----------------------------------------
        :
-filename.ZMD-----------------------------------  ........(8)
TOTAL ZMD SIZE:6,788 bytes                        ........(9)
TOTAL STEPTIME:6912(00001B00)                     .......(10)
TOTAL CHECKSUM:146577(00023C91)                   .......(11)
TOTAL PLAYTIME:00:01:13                           .......(12)
```

(1)トラックヘッダ。これ以下の表示がそのトラックのものであることを示す
(2)そのトラックのステップタイムの合計。「;」を挟んで左側がループ外、右側がループ内の値に対応する。
(3)そのトラックの演奏データ内容のチェックサム
(4)そのトラックに含まれる小節数
(5)小節ヘッダ。これ以下の表示がその小節のものであることを示す
(6)小節内の合計ステップタイム
(7)小節内の演奏データのチェックサム
(8)ZMD名。ただし、表示されない場合もある
(9)ZMDのサイズ
(10)演奏データ全体の合計ステップタイム
(11)演奏データ全体のチェックサム
(12)この演奏データの総演奏時間。コンパイラが計算して算出したものは誤差がある場合がある。

例

```
A>ZP3 -q            (ステータス表示)
A>ZP3 -q music.zms
```

(music.zmsのステータスを表示後、これを演奏する)
(MEASURE3 zm_calc_total(), MEASURE10 ファンクションZM_CALC_TOTAL参照)

■MIDIデータ入出力機能

●MIDIデータの記録

－Aファイルネーム

MIDIのMIDI-IN端子からデータを受信し、これをファイルに出力する。出力フォーマットはスタンダードMIDIファイルのフォーマット0で、これはZ-MUSIC ver.3.0以外でも利用することができる。

ファイル名の拡張子を省略すると'.MID'が自動添付される。

本オプションを使用してZP3.Rを実行すると入力待ちになるので、その後、MIDI機器側のMIDI出力を実行する。受信記録の中断/終了は[ESC]キーで行う。

例

```
A>ZP3 -a data.MID
```

(MEASURE3 zm_midi_rec(), MEASURE9参照)

●MIDIデータの転送

－Xファイルネーム

コマンドオプション'－A'で作成したスタンダードMIDIファイルをMIDI機器へ転送する。拡張子を省略すると'.MID'が自動添付される。

転送の際、Z-MUSIC ver.3.0の状態により、転送を割り込み処理で行う場合と行われない場合とがある。

例

```
A>ZP3 -xU220.MID
```

(MEASURE3 zm_midi_transmission(), MEASURE4 .MIDI_DUMP, MEASURE9参照)

●MIDIインタフェイスの選択

－Dn

MIDIデータのレコーディング「-A」、SMFの転送機能の「-X」、同期演奏機能の「-W」など、MIDIの入出力を使用する場合の使用インタフェイスを明示的に指定するもの。オプションは必ず'-A','-X'よりも先に行わなければならない。

ここで設定されたMIDIインタフェイスはカレントMIDIインタフェイスと設定される。

もし、上に挙げた各機能の実行の際、この指定が行われていないと採択されるインタフェイスは以前のカレントのMIDIインタフェイスとなる。

インタフェイス番号nの設定範囲は1～4で、初期のカレントインタフェイスはインタフェイス番号1。ただし、指定した番号のMIDIインタフェイスが装着されていない場合や、使用できない場合はエラーとなる。

例

```
A>ZP3 -d1 -a data.MID
```

(MIDIインタフェイス1番で受信したデータをdata.MIDで保存する)

```
A>ZP3 -d2 -xU220.MID
```

(MIDIインタフェイス2番にU220.MIDを転送する)
(MEASURE3 zm_current_midi_in(), zm_current_midi_out(), MEASURE4 .CURRENT_MIDI_OUT, .CURRENT_MIDI_IN参照)

●MIDIデータ記録同期演奏

－Y 演奏データファイルネーム,セットアップファイル1,…,セットアップファイル4

'-P'と同じく演奏開始の機能で設定パラメータなどは'-P'と同様のものをとる。演奏直前にMIDIデータの受信準備を行い、演奏と同時に同等のMIDIデータ受信を開始する。受信したMIDIデータはコマンドオプション'-A'の出力と同形式のスタンダードMIDIファイルフォーマット0となる。スタンダードMIDIファイルは演奏したデータの名前とは無関係に必ずZMUSIC.MIDとなる。

主な用途としてはZ-MUSIC用演奏データをスタンダードMIDIファイルに変換する目的で使う。具体的なやり方を下記に示す。

1) X680x0のMIDIインタフェイスのMIDI-OUTをMIDI-INに接続する
2) 'ZP3.R -Y'で希望の演奏データを演奏する

例

```
A>ZP3 -y music.zms
```

3) カレントドライブのカレントディレクトリにZMUSIC.MIDが作成される

●ジュークボックス機能

－Bインデックスファイル,セットアップファイル1,…,セットアップファイル4

－bインデックスファイル,セットアップファイル1,…,セットアップファイル4

'-B'の後ろにインデックスファイル(後述)の名前を書くと、これを読み込む。次に、このファイル中に指定されている演奏データを読み込み、これらを次々に演奏する(ジュークボックス機能)。インデックスファイルの拡張子を省略した場合は、拡張子'.JUK'が自動添付される。

演奏開始指定のスイッチ'-P'と同様に、4つまでセットアップファイルを指定すれば、ジュークボックス開始前に、これらをまず実行することができる。

例

```
A>ZP3 -bINDEX.JUK,MT32SOUND.MDD
```

また、ジュークボックス機能実行中には以下のキー操作が可能となる。

[SHIFT]+[OPT.1]	次の曲へ
[SHIFT]+[OPT.2]	フェードアウトして次の曲へ
[SHIFT]+[CTRL]	演奏の一時停止/再開
[SHIFT]+[XF4]	再演奏
[SHIFT]+[XF5]	フェードアウトして再演奏
[CTRL]+[OPT.1]	前の曲へ
[CTRL]+[OPT.2]	フェードアウトして前の曲へ

●インデックスファイルについて

インデックスファイルは、

　　繰り返し回数(1～255),演奏データファイル名

の書式をとる。ZP3.Rのジュークボックス機能では最大128曲までの演奏データの登録ができ、これらをインデックスファイルに書かれた順番に演奏する。

演奏データはZMD,ZMSの両方とも指定可能。演奏データファイル名の拡張子は省略可能で、省略時は'.ZMD'が自動添付される。

繰り返し回数は省略可能で省略時は繰り返し回数=1が自動設定される。

インデックスファイルに書かれた最後の曲の演奏を完了すると最初の曲へ戻る。(※)

(※)'-b'の小文字のコマンドオプションでジュークボックスを組み込むと、インデックスファイルに書かれた曲を最後まで演奏したあと、初めに戻らずにジュークボックス機能を終了する。

インデックスファイルの例

```
2,ETUDE
3,PRELUDE
FUGE
```

('ETUDE.ZMD'を2ループ演奏し、次に'PRELUDE.ZMD'を3ループ演奏し、最後に'FUGE.ZMD'を1ループ演奏して、これが終了すると初めの'ETUDE.ZMD'へ戻る)

なお、ジュークボックス演奏させることができるZMD/ZMSには以下の制約条件がある。

1) 無限ループは[do]～[loop]命令を使用している場合のみ有効
2) ジュークボックスで演奏させる曲データには、ZPD指定(ZMSコマンド'.ADPCM_BLOCK_DATA'や'.ZPD')以外のAD PCMデータ登録命令が存在してはならない(たとえばZMSコマンド'.ADPCM_LIST'や'.CNF'、(AD)PCM登録ZMSコマンドなど)。もし、含まれていた場合はエラーとなり、ジュークボックス機能は実行されない。

●キー定義の変更

ジュークボックス機能中のキー操作による演奏制御は、環境変数'zp3_jukectrl'にキービットマップ値を列記することによってユーザーの好みにカスタマイズが可能となっている。

set zp3_jukectrl=
<次の曲へ#1, 次の曲へ#2>
<フェードアウト&次の曲へ#1,フェードアウト&次の曲へ#2>
<停止/再開#1, 停止/再開#2>
<再生#1, 再生#2>
<フェードアウト&再生#1,フェードアウト&再生#2>
<前の曲へ#1, 前の曲へ#2>
<フェードアウト&前の曲へ#1,フェードアウト&前の曲へ#2>

キーマップ値の例

[SHIFT]=$E0 [CTRL]=$E1 [OPT.1]=$E2 [OPT.1]=$E3
[XF1]=$A5 [XF2]=$A6 [XF3]=$A7 [XF4]=$B0 [XF5]=$B1

設定例

set zp3_jukectrl=$e2,$b1 $e3,$b1 $e3,$b1 $e3,$e0 $e2,$e0 $e3,$a7 $e0,$a6
↑
「次の曲へ」は[OPT.1]+[XF5]を意味する

●ジュークボックス演奏のインターバルタイムの設定

－Vn

ジュークボックス演奏で1曲を演奏したあと、どのくらいのインターバルタイム(無音時間)を設けるかを設定する。nの設定範囲は0-65535で初期値は480。値が大きいほどインターバルタイムは長くなる。実際のインターバルタイムは、

n×0.016秒

で与えられる(ただし、CPUの負荷状況により多少誤差が出る場合もある)。

なお、設定する場合は必ずオプション'-B'よりも前に設定しなければならない。

例

A>ZP3 -v1000 -bINDEX.JUK

■リアルタイム演奏制御

●リアルタイム演奏制御

－K

このコマンドオプションを設定してZP3.Rを常駐させると、以降、以下のキー操作が可能となる。

[SHIFT]+[OPT.1]	一時停止
[SHIFT]+[OPT.2]	一時停止解除
[SHIFT]+[XF5]	早送り(押している間有効)
[SHIFT]+[XF4]	REPLAY
[SHIFT]+[XF3]	低速演奏(押している間有効)
[SHIFT]+[XF1]	フェードアウト
[SHIFT]+[XF2]	フェードイン

キーと機能の対応は上記のようにデフォルトが定められているが、環境変数'zp3_keyctrl'にキービットマップ値を列記することによってユーザーの好みに変更ができる。

set zp3_keyctrl=<再生#1, 再生#2> <停止#1, 停止#2> <再開#1, 再開#2>
<早送り#1, 早送り#2> <低速#1, 低速#2>
<フェードアウト#1, フェードアウト#2>
<フェードイン#1, フェードイン#2>

キーマップ値の例

[SHIFT]=$E0 [CTRL]=$E1 [OPT.1]=$E2 [OPT.1]=$E3
[XF1]=$A5 [XF2]=$A6 [XF3]=$A7 [XF4]=$B0 [XF5]=$B1

設定例

set zp3_keyctrl=$e2,$b1 $e3,$b1 $e3,$b1 $e3,$e0 $e2,$e0 $e3,$a7
↑
「再生」は[OPT.1]+[XF5]を意味する

【注意】

処理の都合上、ジュークボックス機能(コマンドオプション'-B')との併用はできない(一度後述の'-R'スイッチにて終了する必要がある)。

■常駐解除

●常駐解除

－R

コマンドオプション'-B','-K'設定時はZP3.Rがシステムに常駐するので、機能を停止するためには常駐解除を行わなければならない。コマンドオプションを設定してZP3.Rを実行すると、すでにシステムに常駐しているZP3.Rを常駐解除する。ただしZP3.Rが常駐していないときに実行した場合はエラーとなる。

■同期演奏機能

●同期演奏機能

－W 演奏データファイルネーム,セットアップファイル1,…,セットアップファイル4

コマンドオプション'-P'と同じく演奏開始の機能。設定パラメータなどは'-P'と同様。セットアップファイルの実行後、演奏は開始せず、MIDIデータの$FA(スタートメッセージ)を受信するまで待機状態になる。$FAを受信すると即座に演奏を開始する。

複数のX680x0との同期演奏などを行うときに利用する。

オプション'-W'を利用した複数X680x0の同期演奏

0) 演奏に用いるすべてのX680x0で、Z-MUSICを/Eオプションを設定して常駐する。また使用タイマは統一すること。
A>ZMSC3.X -E
1) X680x0-AのMIDI OUTからX680x0-BのMIDI INへつなぐ
2) X680x0-BのMIDI OUTをMIDI楽器1へつなぐ
3) X680x0-BのMIDI THRUをMIDI楽器2へつなぐ
4) X680x0-Bを、
A>ZP /W filename.ZMS
として演奏待機状態にする
5. X680x0-Aを、
A>ZP filename.ZMS
で演奏開始する

※2台以上の場合はX680x0-?のMIDI OUTをX68-(?+1)のMIDI INへつないでいくか、X680x0-?のMIDI THRUをX680x0-(?+1)のMIDI INへつないでく2通りが考えられるが実用的には4台までが限界と思われる。

■その他

●日本語メッセージの表示

－J

ZP3.Rが出力するメッセージを日本語にする。本コマンドオプションは必ずコマンドラインの先頭に記述しなければ有効とならない。

例

A>ZP3 -j -p etude.ZMS

７．３．　環境変数について

環境変数'zp3_opt'に通常よく設定するコマンドオプションをあらかじめ設定しておくことができる。
たとえば、常にMIDI入出力をインタフェイス番号２で行いたい場合はAUTOEXEC.BATなどに、

```
SET zp3_opt=-D2
```

と設定しておく。そうすれば、

```
A>ZP3 -xU220.MID
```

と実行しただけでも -D2 が追加された、

```
A>ZP3 -D2 -xU220.MID
```

として実行される。

MEASURE 8

ZVT.X

ここでは、サンプリングツール「ZVT.X」の使用法について解説します。

8．1．　ZVT.Xについて

ZVT.XはX680x0のサンプリングデータ((AD)PCMデータ)を扱うツールである。動作は、視覚的に操作が行える「ビジュアルモード」と、Human68kのコマンドライン用外部コマンドとして機能する「コマンドモード」とに分かれる。ここでは各モード別々に解説する。

8．2．　ビジュアルモード

●基本操作

基本的に画面下に表示されているメニューにカーソルを合わせリターンキーもしくはスペースキーを押すことでそれぞれの機能が実行される。メニューには、選択されるとすぐに機能するものと若干のパラメータなどを聞いてくるものに分かれる。このパラメータなどの入力時に[ESC]キーを押すことによってそのメニューの機能の実行を取り消すことができる。

なお、メニューに表されている機能以外にも機能があり、これらはキーボードによるキー入力により実行される（もちろんメニューに表示されているコマンドもキー入力によって実行可）。メニューに表示されている機能は、一部のため、「ZVT.X」を使いこなすためにはキーコマンドを覚える必要がある。

●カーソル操作

カーソル移動にはカーソルキーを使う。

(AD)PCMデータを読み込んだり、サンプリングを実行すると、「START POINT」と「END POINT」の行にデータ列、さらに枠内に波形と青と赤の縦線が表示される。「START POINT」「END POINT」に表示されている数列は、それぞれ枠中の青と赤の縦線が指し示している付近のADPCMデータやPCMデータである。この青と赤の縦線を波形カーソルとよび、波形に対してなんらかの加工処理を行う場合は、この２つの波形カーソルに挟まれた範囲に対して行われる。波形カーソルは、(通常の)カーソルを「START POINT」や「END POINT」に合わせ、テンキーを操作することによって移動させることができる（図１）。

```
7 8 9
4 5 6
1 2 3
0 , .
```

[7] 左へ32ｽﾃｯﾌﾟ移動	[9] 右へ32ｽﾃｯﾌﾟ高速移動
[4] 左へ１ｽﾃｯﾌﾟ移動	[6] 右へ１ｽﾃｯﾌﾟ移動
[1] 左へ波長の1/16ｽﾃｯﾌﾟ移動	[3] 右へ波長の1/16ｽﾃｯﾌﾟ移動
[0] 波形の先頭へ瞬間移動	
[,] 波形の真ん中へ瞬間移動	
[.] 波形の最後尾へ瞬間移動	

図１ 波形カーソルのキー操作

●キーコマンド(アルファベット順)

()内の文字は画面に表示されているメニュー名に対応する。

A
B
アクセス対象バンクをセレクトする。

ZVT.Xには加工したりサンプリングしたりするデータエリアが２つあり、コマンド機能の対象となるデータをそれぞれバンクA、バンクBと呼んでいる。本コマンドでこの操作対象バンクを選択することができる。

C
スタートポイントとエンドポイント間以外の波形を切り捨てる。

D　　(DISK)
ディスクモードへ入る。

1．LOAD
波形データのロードを行う。
ファイルネームを聞いてくるのでパスや拡張子を含めて91文字以内で入力する。ファイル名を省略したり、ワイルドカードを使用するとファイル選択モード(囲み参照)に入る。

2．SAVE
波形データのセーブを行う。
ファイルネームを聞いてくるのでパスや拡張子を含めて91文字以内で入力する。ファイル名を省略したり、ワイルドカードを使用するとファイル選択モード(囲み参照)に入る。

3．SAVE BY 4PHASES
原波形を含む４段階の波形データを自動生成しセーブする(ただし自動生成されるデータはAD PCMデータ形式)。

このモードを選択すると次に、

1．VOLUME　2．PITCH

が現れる。１を選べば４段階のボリュームで、２を選べば４段階の音程でセーブされることになる。

これを選択後さらに、

1．MINUS　2．PLUS

のメニューが現れる。これはどの方向に４段階のデータを生成するかを決めるものである。

ボリュームを選んだ場合にマイナス方向を選択すると原波形のボリュームの、

100%, 88%, 76%, 60%

のデータが自動生成されセーブされる。プラス方向では原波形のボリュームの、

100%, 114%, 126%, 140%

のデータがセーブされる。音程を選んだときには各方向に半音単位の変化量でセーブされる。つまり原波形が今オクターブ４のCだったとすると、マイナス方向では、

O4のC, O3のB, O3のA#, O3のA

プラス方向では、

O4のC, O4のC#, O4のD, O4のD#

がセーブされることになる。

最後にファイルネームを聞いてくるが拡張子は無視され、自動生成される。ファイルの拡張子は強制的に'.4' '.3' '.2' '.1'の４つになる（原波形が'.4'）。

ファイルネームを省略したり、ワイルドカードを使用するとファイル選択モードへと移行する。

4．TEST PLAY
ファイルを音声データと見なして再生する。

ファイルネームを聞いてくるのでパスや拡張子を含めて91文字以内で入力する。ファイル名を省略したり、ワイルドカードを使用するとファイル選択モード(囲み参照)に入る。

ファイル選択モード

ディスク関連のコマンドでファイルネームを省略したり、ワイルドカードを設定するとファイル選択モードになります。ここではカーソルキーによるファイルの選択やディレクトリ間の移動がスムーズに行えるようになっています。
使用キーは以下の通りです。

カーソルキー	カーソルの移動
RETURN/ENTER	ファイルやディレクトリの選択
ESC	ファイル選択モードを終了（ファイル選択の拒否）
TAB	カーソルを左最上部へ瞬間移動。またディレクトリの移動もこれで可能。 スピーディに親ディレクトリに移動可能。
?	ファイルネームの入力。ワイルドカードを指定するとそれに対応したファイルのみが画面に現れる。
SPACE	再生(PCM/AD PCMモードの影響を受ける点に注意)
数字キー(0〜4)	SPACEキーでのPCMデータ再生における再生周波数の切り換え

ファイルの拡張子とデータ形式

拡張子'.PCM'のファイルはZVT.XではAD PCMファイルと自動的に見なしてロード/セーブします。また拡張子'.P16'のファイルは16ビットPCMファイルと見なしてロード/セーブします。それ以外の拡張子を持ったファイルをアクセスする場合は、その時点でのデータモードがAD PCMならAD PCMとして、データモードが16ビットPCMデータなら16ビットPCMデータとしてアクセスします。なるべく'.PCM'、'.P16'のどちらかを用いるようにしましょう。

E

PCMデータおよびAD PCMデータにエフェクト(加工)をかける。
1.↑/↓　2.←/→　3.ADD 0 TO TAIL　4.REVERSE　5.CHORUS
6.DELAY　7.SFX　8.SMOOTHING　9.ENVELOPE
の9種類のエフェクトよりひとつを選択する。

1．↑/↓
PCMデータをグラフのY軸方向にシフトするもの(マイナス値で下へ)機能選択後、どのくらいシフトするか(オフセット値)を聞いてくる

2．←/→
PCMデータをグラフのX軸方向にシフトするもの(マイナス値で左へ機能選択後、どのくらいシフトするか(オフセット値)を聞いてくる

3．ADD 0 TO TAIL
データの最後尾にn個の0を加える
機能選択後、オフセット値を聞いてくる

4．REVERSE
逆転

5．CHORUS
音が透き通った感じになる

6．DELAY
エコーがかかったような音になる

7．SFX
金属音に変化する

8．SMOOTHING
こもったような音に変化する

9．ENVELOPE
エンベロープを変形する(フェードイン/アウト)
機能選択後、イン/アウトレベル(0〜127)とエンベロープの変更方向を聞いてくる
1．＜ … フェードイン
2．＞ … フェードアウト

設定したパラメータとその効用は図2参照
なお、機能は青赤の2つの波形カーソルに挟まれた区間に対して働くことに注意。

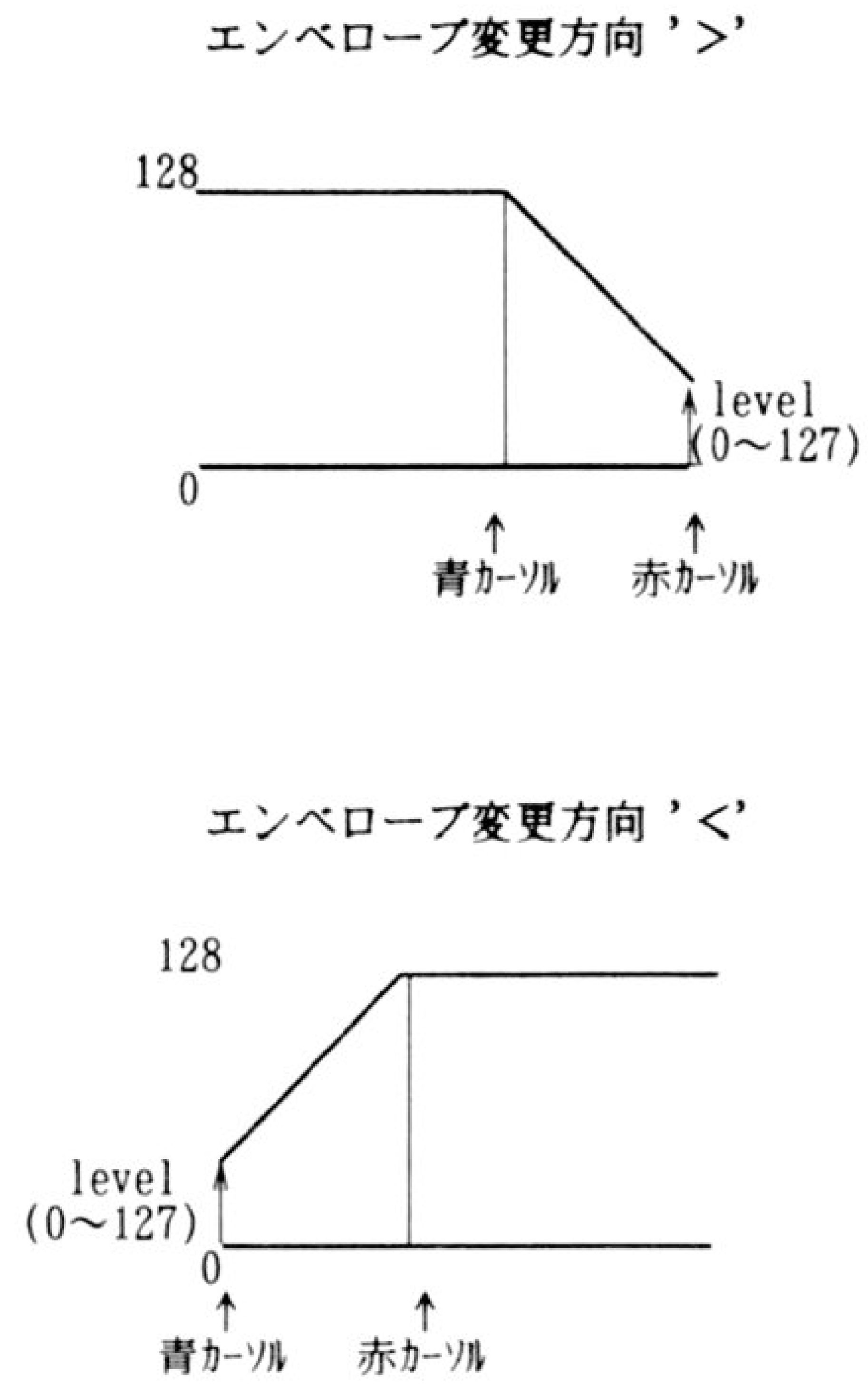


図2　エンベロープ変更方向とイン/アウトレベル

オフセット値について

「←/→」と「ADD 0 TO TAIL」では、機能メニューを選択後、オフセット値を聞いてきますがこれはデータモードによって効用が異なります。
たとえばPCMモードのとき1000と入力するとオフセットはPCMデータ1000個分(つまり2000バイト)ということになります。またAD PCMモードで1000と入力した場合にはAD PCMデータ1000個分(つまり1000バイト)ということになります。

オフセット値の例

データモードがPCMのとき

3.9kHz	1秒＝3900
5.2kHz	1秒＝5200
7.8kHz	1秒＝7800
10.4kHz	1秒＝10400
15.6kHz	1秒＝15600

データモードがAD PCMのとき

3.9kHz	1秒＝1950
5.2kHz	1秒＝2600
7.8kHz	1秒＝3900
10.4kHz	1秒＝5200
15.6kHz	1秒＝7800

「↑/↓」のオフセット値はPCMデータへ直接加算される数値を意味しているので、演算はデータモードにはまったく無関係にPCMデータ次元で行われます。

F　(SAMPLING RATE)

サンプリング時や再生時の周波数の設定をする。サンプリング周波数は一般に、周波数が高ければ高いほど高音質となる。

I

波形の頭と後尾を見つける。単純なアルゴリズムによる探査のため複雑な波形に対しては希望の結果を得られないことがある。また、データモードによって動作のしかたが微妙に違うため、データモードを変えて実行すると違った結果を得ることができる。

J

音程の変更を1Hz単位で行える。コマンド入力後、変更前の周波数と変更後の周波数を聞いてくる。周波数パラメータ設定範囲は1～65535。変換前と変換後の周波数は厳密に入力する必要はなく、両周波数の整数比を入力するだけでもよい。つまり、「1オクターブ上げる(周波数を2倍にする)」場合、

変更前周波数=1Hz　　変更後周波数=2Hz

でも、

変更前周波数=100Hz　変更後周波数=200Hz

でも得られる結果は同じである。

K　　(PITCH)

半音単位でピッチをシフトする。パラメータは－12から＋12。±1オクターブの音程変換が可能。

M　　(MONITOR)

モニタ機能のオン/オフ。「AUDIO IN」端子からの入力をX680x0のA/D、D/Aコンバータを通してヘッドホンなどで聞くことができる。

N

ポルタメント(音程を滑らかに変化させる)を1Hz単位で行う。コマンド入力後、その音の基本周波数と変化希望周波数を聞いてくる。周波数パラメータ設定範囲は1～65535。2つの周波数パラメータは厳密に入力する必要はなく、両周波数の整数比を入力するだけでもよい。つまり、「1オクターブ上へ滑らかに変化させる(周波数を滑らかに2倍にする)」場合、

変更前周波数=1Hz　　変更後周波数=2Hz

でも、

変更前周波数=100Hz　変更後周波数=200Hz

でも得られる結果は同じである。

国際基準イ＝a1＝440Hzに基づく12平均率音階周波数表

（試験に出るX1より転載）

	00	01	02	03	04	05	06	07	08
C	16.352	32.703	65.406	130.81	261.63	523.25	1046.5	2093.0	4186.0
C＃	17.324	34.648	69.296	138.59	277.18	554.37	1108.7	2217.5	4434.9
D	18.354	36.708	73.416	146.83	293.66	587.33	1174.7	2349.3	4698.6
D＃	19.445	38.891	77.782	155.56	311.13	622.25	1244.5	2489.0	4978.0
E	20.602	41.203	82.407	164.81	329.63	659.26	1318.5	2637.0	5274.0
F	21.827	43.654	87.307	174.61	349.23	698.46	1396.9	2793.8	5587.7
F＃	23.125	46.249	92.499	185.00	369.99	739.99	1480.0	2960.0	5919.9
G	24.500	48.999	97.999	196.00	392.00	783.99	1568.0	3136.0	6271.9
G＃	25.957	51.913	103.83	207.65	415.30	830.61	1661.2	3322.4	6644.9
A	27.500	55.000	110.00	220.00	440.00	880.00	1760.0	3520.0	7040.0
A＃	29.135	58.270	116.54	233.08	466.16	932.33	1864.7	3729.3	7458.6
B	30.868	61.735	123.47	246.96	493.88	987.77	1975.5	3951.1	7902.1

O　　(PANPOT)

再生やモニタ機能のパンポットを設定する。

P　　(PLAY)

バンクセレクトされているバンクのデータを再生する。

Q　　(EXIT)

[ESC]＋[Q]でZVT.Xを終了する。誤操作を避けるためこのコマンドのみ2キーコマンドになっている。

R　　(RECORD)

録音を行う。オートサンプリング機能のスイッチ（TRIGGER）がオン、さらにモニタ機能（MONITOR)オンのときにはオートサンプリングを行うがこれらのスイッチが片方でもオフのときには通常のサンプリングを行う。

S　　(DATA SIZE)

サンプリングする音の時間をデータサイズによって設定する。

T　　(SAMPLING TIME)

サンプリングする音のデータサイズを「秒」で設定する。

V　　(VOLUME)

音量を変化させる。パラメータはパーセント単位。1～300%の範囲のみ有効。

X

アクセスバンクの切り換え。[X]キーを押すたびにセレクトバンクが、A/Bと交互に切り換わる。

Z

2つのデータ間の畳み込み演算を行う。2つの音声データの特性をひとつにまとめたりすることができる。演算結果をどちらのバンクに格納するかを聞いてくる。

1．A＊B→B … 演算結果をBへ格納する
2．A＊B→A … 演算結果をAへ格納する

なお、演算にはかなりの時間を必要とする。

@　　(TRIGGER)

オートサンプリング機能の有効/無効を設定する。

＊

オートサンプリングとは

オートサンプリングとは音が入力されたことを自動的に感知して録音を開始する機能のことです。ZVT.Xではオートサンプリング機能のスイッチ（TRIGGER）とモニタ機能（MONITOR)の両方がONのときに録音キーを押すとこの機能が働きます。どちらかがOFFだとこの機能は働きません。オートサンプリングモードでないときは録音キー[R]が押された瞬間から録音を開始します。

オートサンプリングの音の入力待ちのときは一切のキー入力を受け付けません。なにか、音の入力があるまで音を待ち続けます。

ZVT.Xのオートサンプリング機能動作フローチャート

[R]キーを押す
↓
音の入力があるまでひたすら待つ
↓
音が入力されると実際に録音を開始する

PCMのデータ表示を10進数または16進数へ交互に切り換える。デフォルトは16進数モード。

＋　　(COMPOUND)

バンクAとBのデータをミキシングしてひとつのデータにする。パラメータとしてミックス方向を聞いてくる。

1)　A→B … AとBをミックスして結果はBへ格納する
2)　A←B … AとBをミックスして結果はAへ格納する

－

バンクAとBのデータ間の算術/論理演算を行う。演算子を、

1.ADD　　和
2.SUB　　差
3.OR　　論理和
4.AND　　論理積
5.EOR　　排他的論理和

の5つから選択する。演算子タイプ選択後、演算結果をどちらのバンクに格納するのかを聞いてくる。

1．A○B→B … 演算結果をBへ格納する
2．A○B→A … 演算結果をAへ格納する

／

ソースバンクのデータをデスティネーションバンクのデータのスタートポイントへはめ込む。パラメータとしてはめ込む方向を聞いてくる。

1．A→B … AをBのスタートポイントへはめ込む
2．A←B … BをAのスタートポイントへはめ込む

＝

バンク間のデータのコピー。パラメータとしてコピー方向を聞いてくる。
1．A→B … AをBへコピーする
2．A←B … BをAへコピーする

?
各バンクの情報を表示する。
ADPCM DATA ADDRESS:AD PCMデータが格納されているアドレス
START POINT:青カーソルのオフセット位置
END POINT:赤カーソルのオフセット位置
ADPCM DATA SIZE:AD PCMデータの大きさ
EFFECTIVE DATA SIZE:青/赤カーソルに挟まれた区間の大きさ
PCM DATA ADDRESS:PCMデータが格納されているアドレス
FREE AREA:フリーエリア

CLR
セレクトされているバンクのデータをクリアする。

TAB
データモード(PCM/AD PCM)の切り換えを行う。

8．3．　コマンドモード

Human68kのコマンドラインからZVT.Xの機能を利用することができ、この動作モードを特に「コマンドモード」と呼んでいる。なお、コマンドモードにおけるZVT.Xが取るコマンドラインは以下の２形式となる。
1) ZVT <ｺﾏﾝﾄﾞ・ｽｲｯﾁ列1> <ﾌｧｲﾙﾈｰﾑ1> <ﾌｧｲﾙﾈｰﾑ2>
2) ZVT -M[n] [ｺﾏﾝﾄﾞ・ｽｲｯﾁ列1] <ﾌｧｲﾙﾈｰﾑ1> [ｺﾏﾝﾄﾞ・ｽｲｯﾁ列2] <ﾌｧｲﾙﾈｰﾑ2> <ﾌｧｲﾙﾈｰﾑ3>

ファイルネーム１で指定された(AD)PCMデータファイルを読み込み、これをコマンドオプションで指定された機能で加工処理を行い、これをファイルネーム２としてセーブする…これがコマンドモード1)の形式の基本プロセスとなる。
コマンドオプション'-M'を使用した、ミキシング(合成)処理を実行するときのみファイルネーム３の指定を必要とし、これが2)の形式となる。このときはまず、ファイルネーム１で指定された(AD)PCMデータファイルを読み込み、これをコマンドスイッチ列１で加工する。次にファイルネーム２で指定された(AD)PCMデータファイルを読み込み、これをコマンドオプション列２で加工し、これら２つを合成してファイルネーム３でセーブする……というプロセスになる。

8．3．1．ZVT.Xのコマンドオプション

－４Ｐ＋
－４Ｐ－
ビジュアルモードの４段階データセーブに相当するもの。ファイルネーム１のAD PCMデータを読み込み、これに対して半音単位で４段階のピッチシフトを行い拡張子'.4','.3','.2','.1'を添付したファイルネーム２でセーブする。'－４Ｐ'の後ろの＋,－の符号はピッチシフトの方向を表す。＋が音程を４段階に上昇させる指定で、逆に－が音程を４段階に下降させる指定となる。
たとえば、FILE1.PCMがオクターブ４のCだったときに、
A>ZVT -4p+ FILE1.PCM ABC
とすると、
オクターブ４のC(原波形)が'ABC.4'
オクターブ４のC#が'ABC.3'
オクターブ４のDが'ABC.2'
オクターブ４のD#が'ABC.1'
としてセーブされる。

－４Ｖ＋
－４Ｖ－
ビジュアルモードの４段階データセーブに相当するもの。ファイルネーム１のAD PCMデータを読み込み、これに対して％単位で４段階の音量変更を行い拡張子'.4','.3','.2','.1'を添付したファイルネーム２でセーブする。'－４Ｖ'の後ろの＋,－の符号は音量変更の方向を表す。＋が音量を４段階に上昇させる指定で、逆に－が音量を４段階に下降させる指定となる。
たとえば、
A>ZVT -4v+ FILE1.PCM ABC
とすると、
FILE1.PCMの音量の100%(原波形)が'ABC.4'
FILE1.PCMの音量の114%が'ABC.3'
FILE1.PCMの音量の126%が'ABC.2'
FILE1.PCMの音量の140%が'ABC.1'
としてセーブされる。
また、
A>ZVT -4v- FILE1.PCM ABC
とすると、
FILE1.PCMの音量の100%(原波形)が'ABC.4'
FILE1.PCMの音量の88%が'ABC.3'
FILE1.PCMの音量の76%が'ABC.2'
FILE1.PCMの音量の60%が'ABC.1'
としてセーブされる。

－Ａ
ファイルネーム１の16ビットPCMデータをファイルネーム２のAD PCMデータへ変換する。

－Ｃ
ファイルネーム１のAD PCMデータをファイルネーム２の16ビットPCMデータへ変換する。

－Ｇ
基本的にビジュアルモードはG-RAMがRAMディスクなどで使用されていると起動不可能である。しかし、このコマンドオプションを指定するとG-RAMディスクなどを破壊し強制的にビジュアルモードへ移行させることができる。

－Ｉn
ファイルネーム１のAD PCMデータをファイルネーム２のAD PCMデータへはめ込みファイルネーム３でセーブする。－Ｉの後ろにオフセット値を記述できるが、これはAD PCMのバイト数単位で指定する(通常は0または省略で構わない)。

－Ｌ filename1, filename2
ビジュアルモード起動時にfilename1をバンクＡに、filename2をバンクＢに読み込んで起動する。起動時に読み込める音声ファイルはAD PCMに限定される。

－Mn
ファイルネーム１のAD PCMデータとファイルネーム２のAD PCMデータを、加工指定があればそれぞれ加工処理し、これらをミックス(合成)しファイルネーム３でセーブする。－Ｍの後ろにオフセット値を書けるがこれはAD PCMのバイト数単位で指定する(通常は0または省略で構わない)。

－Pn
ファイルネーム１のAD PCMデータの音程をn半音分シフトしファイルネーム２でセーブする。パラメータnは－12から＋12まで設定可能。

－Vn
ファイルネーム１のAD PCMデータの音量を1%単位で変更し、ファイルネーム２でセーブする。パラメータnは1～300%まで設定可能。

8．3．2．コマンドオプションの複数設定について

コマンドオプションの'Ｐ','Ｖ'は同時に設定可能である。また、コマンドオプション'Ｐ','Ｖ'はコマンドオプションの'４','Ａ','Ｃ','Ｍ'と同時に設定可能である。
以下に使用具体例をいくつか示す。
●例１
A>ZVT -p3 -v150 ABC.PCM DEF.PCM
'ABC.PCM'の(AD)PCMデータを＋3半音分シフト、音量を150%にしたものを'DEF.PCM'でセーブする。
●例２
A>ZVT -a -p+5 ABC.PCM DEF.PCM
'ABC.PCM'のＰＣＭデータを5半音分上へシフトした上でAD PCMデータへ変換し、これを'DEF.PCM'でセーブする。
●例３
A>ZVT -c -p-5 ABC.PCM DEF.P16 -v180
ファイルネームの後ろにもコマンドオプションは設定可能である。この

例では'ABC.PCM'のAD PCMデータを－5半音分シフトし、さらに音量を原波形の180%に変更し16ビットPCMデータで'DEF.P16'でセーブする。

●例4

A>ZVT -p-3 -v50 ABC.PCM -p5 -v120 DEF.PCM GHI.PCM -m

'ABC.PCM'のAD PCMデータを－3半音分シフト、音量を50%にしたものと'DEF.PCM'のAD PCMデータを＋5半音分シフト、音量を120%にしたものとをミックスし'GHI.PCM'でセーブする。

●例5

A>ZVT -4v+ -v80 -p-7 ABC.PCM DEF

'ABC.PCM'のAD PCMデータを－7半音分シフトし、音量を、

80%×100%＝80%に変更したものを'DEF.4'でセーブする。
80%×114%＝91.2%に変更したものを'DEF.3'でセーブする。
80%×126%＝100.8%に変更したものを'DEF.2'でセーブする。
80%×140%＝112%に変更したものを'DEF.1'でセーブする。

8．3．3．ワイルドカードについて

ファイルネームにワイルドカードが使用可能である。たとえば、

A>ZVT -v150 -p5 *.PCM *1.PCM
A>ZVT -m *.PCM A*.PCM *B.PCM

といった使い方が可能。ただし、ワイルドカードの組み合わせによってはエラーとなる場合もある。

8．3．4．ファイルネーム'PCM'について

Z-MUSIC ver.3.0またはPCMDRV.SYSが組み込まれているとき、出力側のファイルネームに'PCM'を設定すると、生成されたデータはファイルとして生成されずに、直接再生することができる。

たとえば、

A>ZVT -p-3 -v50 ABC.PCM -p5 -v120 DEF.PCM PCM -m

とすると'ABC.PCM'の(AD)PCMデータを－3半音分シフト、音量を50%にしたものと'DEF.PCM'の(AD)PCMデータを＋5半音分シフト、音量を120%にしたものとをミックスしこれを保存せずに音声として聞くことができる。

MEASURE 9

スタンダードMIDIファイルとローランドエクスクルーシブ

ここではZ-MUSIC ver.3.0の機能のひとつであるMIDI機器の設定をファイルとしてセーブする機能について解説します。また、MIDI機器のメモリを操作するエクスクルーシブメッセージについて具体的な用例を挙げて解説します。

9．1．　MIDI機器のメモリデータをセーブする

Z-MUSIC ver.3.0では常駐後に使用可能となるMIDI受信機能を用いてMIDI機器の状態（音色や設定など）をファイルに保存することができる。これを利用すればMIDI楽器の設定状態や音色のデータをX680x0上で管理することが可能となる。ここではこの機能の具体的な使用方法について述べる。

まず、MIDI機器側にMIDI OUT端子がなくてはならない。さらにMIDI機器側に機器自身のメモリ内データを外部へ送信する機能が必要となる。MIDI OUT端子があるような楽器ならばたいていは備わっている機能であるので楽器のマニュアルを参照してほしい(たいていはマニュアルの「MIDIデータダンプ」「バルクダンプ」といった項目に載っているはずである)。

次にX680x0のMIDI IN端子とMIDI機器のMIDI OUT端子をMIDIケーブルで接続する。

以上の準備が前提となる。

1) ZMSC3.Xの常駐を確認する
2) 楽器側のバルクダンプのメニューを出し確認する
3) コマンドオプション'-A'を使用してZP3.Rを実行する
　　A>ZP3 -A filename
4) 楽器のバルクダンプを実行する
5) 楽器のコンソールに"COMPLETED"などのメッセージが表示されるのを確認するまで待つ
6) [ESC]キーを押してZP3.RのMIDI受信モードを終了させる
7) '-A'の後ろに指定したファイルネームでフォーマット０のスタンダードMIDIファイルが出力される

9．2．　スタンダードMIDIファイルの楽器への転送

9.1.の手順で作成したスタンダードMIDIファイル(以下、SMF)を、MIDI機器側へ転送する方法について解説する。

●Human68kのコマンドラインから行う場合（その１）

1) ZMSC3.Xの常駐を確認する
2) Human68kの内部コマンド「COPY」を用いて希望のSMFをシステムファイル名MIDIに転送する。

例

```
A>copy filename.MID MIDI
```

●Human68kのコマンドラインから行う場合（その２）

1) ZMSC3.Xの常駐を確認する
2) ZP3.Rにコマンドオプション'-X'を設定して目的のファイルを送信する。'-X'の後ろに転送したいSMFのファイル名を記述する。(出力MIDIインタフェイスを指定したい場合はZP3.Rのコマンドオプション'-D'を使用する。(MEASURE7参照)

例　(出力MIDIインタフェイス番号が2、転送したいSMFのファイル名が"filename.MID"の場合)

```
A>ZP3 -D2 -X filename.MID
```

●X-BASICから行う場合

1) ZMSC3.Xの常駐を確認する
2) MUSICZ3.FNCを組み込んだX-BASICを起動する
3) 関数zm_transmit_midi_dump()を用いて転送する

例　(出力MIDIインタフェイス番号が2、転送したいSMFのファイル名が"filename.MID"の場合)

```
zm_transmit_midi_dump( 2,"filename.MID")
```

●ZMSから行う場合

ZMSコマンド「.SMF」を用いる。

例　(出力MIDIインタフェイス番号が2、転送したいSMFのファイル名が"filename.MID"の場合)

```
.SMF 2,filename.MID
```

スタンダードMIDIファイルとは

スタンダードMIDIファイルにはフォーマット０，１，２の３種類のフォーマットがあります。Z-MUSIC ver.3.0で取り扱えるのはフォーマット０です。簡単に各フォーマットの違いを述べておきます。

フォーマット0はパートの概念がないフォーマットで、演奏情報は１本の時間軸の中で表されます。MIDIケーブルを通ってきた情報をそのまま時間情報を加えて記録した、というイメージです。音色データなどはこのフォーマットで管理されることが多いようです。複数チャンネルで構成されるフォーマット０の演奏データも存在します。そういった演奏データはコンバータソフトウェアを使いフォーマット１に変換することもできます。

フォーマット1はもっとも頻繁に用いられるフォーマットで、演奏データは複数のパートで管理されています。トラック(パート)の概念を持った市販のDTMツールはたいていこのフォーマットを読み書きできるようです。パートの概念があるためデータの応用性は高いといえます。

フォーマット２は複数のテンポを持ったトラックが共存できるものであまり一般的ではないようです。

イメージ的にはフォーマット０がたくさん集まってフォーマット１、フォーマット１がたくさん集まってフォーマット2と捉えるとわかりやすいでしょう。

9．3．　ローランドエクスクルーシブについて

多くのユーザーがローランド製のMIDI機器を使用していると思われるが、ここではそのローランド製のMIDI機器のエクスクルーシブメッセージの使用方法について解説を行う。

Z-MUSIC ver.3.0では、X-BASICではMUSICZ3.FNC命令である、

　　zm_exclusive()

が、ZMSコマンドでは、

　　.ROLAND_EXCLUSIVE

が、MMLでも、

　　X

といった専用命令が用意されており、容易にMIDI機器のメモリを書き換えることが可能となっている。

9．3．1．　マニュアルの後ろを見てみる

ここではZMSコマンド .ROLAND_EXCLUSIVE を使ってCM-32P(あるいはCM-64のPCM音源パート)のパーシャルリザーブの行い方を例としてローランドエクスクルーシブの使用方法を解説する。

まずCM-64のマニュアル33ページを開いてみる。'＊６－３'という表に

注目する。この表の中段に、

```
| 00 04 | 00aa aaa | PARTIAL RESERVE (PART 1) 0-31 |
| 00 05 | 00aa aaa | PARTIAL RESERVE (PART 2) 0-31 |
| 00 06 | 00aa aaa | PARTIAL RESERVE (PART 3) 0-31 |
                    :
                    :
| 00 09 | 00aa aaa | PARTIAL RESERVE (PART 6) 0-31 |
```

が見つかるはずである。これを確認したら、次にこの表'＊6－3'のタイトルを見る。すると、

ＳＹＳＴＥＭ　ＡＲＥＡ

とある。ここで、この表の右にあるアドレスマップからこの'SYSTEM AREA'という文字列を捜す。すると、

52 00 00

が'SYSTEM AREA'のベースアドレスということがわかる。つまり、表'＊6－3'の中のパラメータを設定したい場合は、表の左に書いてあるオフセットアドレスとこのベースアドレスを足したアドレスに値を送信すればよいことになる。この例でいくと、

52 00 00 + 00 04 = 52 00 04

がパーシャルリザーブのパラメータのアドレスということになる(表'＊6－3'のタイトルの下を見ると「パーシャルリザーブは6パート分を一度に送らなければ無効です」とあるのでもちろんこれに従うこと)。

結局、以下のようにコマンドを書けばよいことになる。

```
.ROLAND_EXCLUSIVE dev,mdl {    $52,$00,$04
                               2
                               3
                               5
                               4
                               3
                               6}
```

(1パラメータごとに改行しているが特に意味はない。$52,00,04がアドレス、それ以降がパラメータ)

この例では、パート1～6、それぞれ2,3,5,4,3,6声をリザーブしている。もちろんそのMIDI機器の最大同時発声数を超えてはならない。この例では2+3+5+4+3+6＝23で、まだ31－23＝8声分の余裕があることがわかる。

さて今の例で、dev,mdlというのが出てきたが、これは操作対象の楽器のID番号のことを表している。

mdlはモデルIDと呼ばれるものでメーカーがその楽器に与えた識別番号のこと。MT-32/CM-32L/CM-32P/CM-64はすべて$16と決められている。CM-64ならばマニュアルの28ページの上部に記載されている。

devはデバイスIDと呼ばれるもので、同じ機種(あるいは同一のデバイスIDを持ったMIDI機器)でも、これを変えておけば個別に操作が可能となる、ユーザーが任意に設定可能な識別番号である。しかしCM-32L/CM-32P/CM-64は$10に固定であるため、結局、上の例は、

```
.ROLAND_EXCLUSIVE $10,$16 {    $52,$00,$04
                               2
                               3
                               5
                               4
                               3
                               6}
```

ということになる。

以上の手順をまとめると、

1)　設定したいパラメータの表を探す
2)　表のタイトルをアドレスマップより探す
3)　設定したいパラメータのオフセット値と2)でみつけたベースアドレスを加算する
4)　2種類のIDを確認してコマンドを書く

となる。

9．3．2．　その他の例

CM-32P(あるいはCM-64のPCMパート)の各パートの受信MIDIチャンネルを設定するときは、

```
ROLAND_EXCLUSIVE $10,$16 {$52,$00,$0a,0,1,2,3,4,16}
```

のようにする。この例ではパート1～5をそれぞれチャンネル1～5へ、パート6はOFFにしている。MIDIチャンネルは通常1～16ということになっているが内部処理的には0～15なので注意すること。そういうわけでパラメータの0はMIDIチャンネル1、パラメータの1はMIDIチャンネル2……以下同様ということになる。最後のパラメータの16はMIDIチャンネル17ということではなくそのパートをミュートする(使用しない)ということを意味する。多くのローランド製のMIDI機器は、

MIDIチャンネル	実際のパラメータ値
1～16	0～15
OFF	16

ということになっている。

同様にしてリバーブの設定や各パートの細かい設定(リバーブのON/OFF，キートランスポーズの設定など)も行うことができる。もちろん、X-BASIC上でMUSICZ3.FNC命令の'zm_exclusive()'や、MMLの'X'コマンドを用いても、いままで述べてきた同様な方法で行うことができる。

注意すべきなのは、MIDI機器上のパネルで1～128で使用していた数値パラメータが、ローランドエクスクルーシブを用いて設定する場合には0～127として設定しなくてはならないことがある点。ほかにも-64～+63で使用していたものが、0～127にシフトされていたりする。よってパラメータ数値を決定する場合には、事前に機器上でのパラメータと内部的なパラメータとの対応を十分理解しておく必要がある。

エクスクルーシブ送信後のウェイト

Z-MUSIC ver.3.0では、スタンダードMIDIファイル(SMF)の転送時などにおいてEOX(END OF EXCLUSIVE MESSAGE)$F7を送信した場合、デフォルトではその直後、一定時間のウェイトを入れるようになっています。これはMIDI機器側がエクスクルーシブメッセージを処理するのに多少の時間を必要とするためです。このウェイトの値はZMSC3.X常駐時のコマンドオプション'-X'で設定することができます(MEASURE2参照)。

SMFの演奏データ等を演奏する場合、逆にこの機構が、余計にウェイトを入れすぎて、演奏のテンポを不安定にしてしまう場合があります。その場合は、おなじくZMSC3.X常駐時のコマンドオプション'-X'で0を設定し、ウェイトを無くしてみてください。

ところで、MMLの@XやXコマンドを用いてエクスクルーシブメッセージを送信した場合は、音楽演奏のテンポずれを避けるために('-X'の設定によらず)、このウェイト処理を実行しません。このようなケースでは、休符などを使ってユーザーが自前でEOX送信後のウェイトを与えてやる必要があります。

MEASURE 10

Z-MUSIC ver.3.0のファンクションコール

ここではZMSC3.XおよびZMC.Xが常駐したあとに使用できるファンクションコールについて説明します。
使用にはMPU680x0の機械語の知識が必要です。
Z-MUSICを使って音楽演奏を楽しむだけの方は読み飛ばしてしまって構いません。

10．1．　ファンクションコールのコール法

Z-MUSIC ver.3.0のファンクションコールの仕方は以下のような方法で行う。

```
        moveq.l #n,d0           *n=ファンクションナンバー
        lea     ～,a1           *パラメータが必要であれば設定
        moveq.l #m,d1           *パラメータが必要であれば設定
        trap    #3
```

戻り値はd0.lとa0.lのみで、その他のレジスタは全部保存される。ただし、逆に戻り値がない関数においても、d0.lとa0.lの内容はファンクションコール前と異なる場合がある(保存されない)点に注意。

ファンクションにもよるが通常はd0.l=0が正常終了を意味する。万が一、ファンクション内でエラーが発生した場合のエラーコードは($8000+エラーの発生したファンクション番号(0-32767))*65536+エラーID(0-65535)がd0.lに代入される。またこのときはa0.lの内容は０になる。なお、エラーIDとその説明に関してはMEASURE11を参照のこと。

ZMSC3.Xの常駐チェック方法はリスト１の様に行う。TRAP #3のベクタが指し示すアドレス－8から文字列"ZmuSiC"があるかどうかを調べている。さらに"ZmuSiC"の後ろにはバージョンコードが2バイトが格納されている。バージョンコードは、

上位バイト上位4ビット:バージョン整数部
上位バイト下位4ビット:バージョン少数第1桁
下位バイト上位4ビット:4
下位バイト下位4ビット:バージョン少数第2桁

となっている。

例

VERSION 3.21 → "ZmuSiC",$32(.b),$41(.b)

リスト１

```
chk_drv:
        * > eq=常駐確認
        * > mi=常駐していない
        move.l  $8c.w,a0
        subq.w  #8,a0
        cmpi.l  #'ZmuS',(a0)+
        bne     chk_drv_err
        cmpi.w  #'iC',(a0)+
        bne     chk_drv_err
        cmpi.w  #$3000,(a0)+    *バージョンが古い
        bcs     chk_drv_err
        moveq.l #0,d0
        rts
chk_drv_err:
        moveq.l #-1,d0
        rts
```

10．2．　ファンクションコール解説

●数値表現について

・(.l),(.w),(.b)はそれぞれ32bit,16bit,8bitの値を表す。
・d0.lwはd0の下位16ビット値、d0.hwはd0の上位16ビット値を意味する。
・d0.lbはd0の第0～7ビット値、d0.hbはd0の第8～15ビット値を意味する。
・d0.hwhb,d0.hwlb,d0.lwhb,d0.lwlbはそれぞれd0の最上位8ビット、第16～23ビット、第8～15ビット、最下位8ビットを表す。
・例で用いているマクロZ_MUSICは、

```
        Z_MUSIC macro   number
                moveq.l number,d0
                trap    #3
                endm
```

のようになっている。

ファンクション$00　ZM_INIT

機能　Z-MUSICの初期化を行う
引数　d1.l:0(dummy)
戻り値　d0.l:Z-MUSICのバージョンID
備考　d1.lは将来の拡張を考慮して現時点ではかならず0を設定する。
バージョンIDはたとえばバージョン3.21ならば $00003241となる。
４の部分は固定値。

ファンクション$01　ZM_ASSIGN

機能　チャンネルアサイン
引数　d1.hw:デバイスタイプ(0=FM音源,1=AD PCM音源,$8000=MIDI1,$8001=MIDI2,$8002=MIDI3,$8003=MIDI4,-1=カレントMIDI)
d1.lw:チャンネル番号0-15
d2.l:トラック番号0-65534
戻り値　d0.l=0:正常終了
d0.l=0以外:エラーコード
備考　カレントMIDIはMIDIインターフェース番号を指定しない場合のデフォルトとなるものでファンクションZM_CURRENT_MIDI_OUTで設定する。

ファンクション$02　ZM_COMPILER

機能　ZMSをコンパイルする
引数　d1.lのd0-d6:検出するエラーの最大値/0:∞,1-127
d1.lのd15:エラーテーブルリストを作成して出力するか
(0:no,1:yes)
d2.l:ZMSサイズ
a1.l:ZMSが格納されているアドレス
戻り値　d0.l=0:正常終了
a0.l=ZMDのアドレス(コンパイル中にウォーニングが発生した場合はZMDの後ろにくっつく)
d0.l=0以外:コンパイルで発生したエラーの数
a0.l=エラーテーブル
備考　・エラーテーブルの構成についてはMEASURE11を参照のこと。
・エラーテーブルは使用後ファンクション FREE_MEMなどで解放すること。
・d2.l,a0.l共に0にするとコンパイラの常駐解除となる。通常は指定してはならない。正常に解除した場合の戻り値はd0.l=0、失敗時はd0.l=エラーコードが返される。
・コンパイラは外部組み込みのファンクションであるため、使用前にはコンパイラの常駐を確認する必要がある。

コンパイラの常駐確認方法

```
.include  zmcall.mac
moveq.l #ZM_COMPILER,d1
```

```
lea      -1.w,a1
Z_MUSIC  #ZM_HOOK_FNC_SERVICE
move.l   a0,d0
bne      コンパイラあり
コンパイラなし
```

ファンクション$03　ZM_VGET

機能　FM音源音色を取り出す
引数　d1.l:音色番号(0-32767)
　　　　d2.l:0=OPMDRV.X互換, 1=AL/FB分離方式
　　　　a1.l:音色データ格納バッファ
戻り値　0:正常終了
　　　　0以外:エラーコード
備考　a1.lで表される音色データの書式は以下のとおり。
　　　　(0～54で表される数値はそれぞれ0(a1)～54(a1)を意味する)

●音色フォーマット(OPMDRV.X互換方式)

```
0 :AF(ﾌｨｰﾄﾞﾊﾞｯｸ/ｱﾙｺﾞﾘｽﾞﾑ) (0～63)
1 :OM(ｽﾛｯﾄﾏｽｸ)          (0～15)
2 :WF(ｳｪｰｳﾞﾌｫｰﾑ)         (0～3)
3 :SYC(ｼﾝｸﾛ)             (0,1)
4 :SPD(ｽﾋﾟｰﾄﾞ)           (0～255)
5 :PMD                   (0～127)
6 :AMD                   (0～127)
7 :PMS                   (0～7)
8 :AMS                   (0～3)
9 :PAN                   (0～3)
10:DUMMY
11(OP1),22(OP2),33(OP3),44(OP4):AR(ｱﾀｯｸﾚｰﾄ)          (0～31)
12(OP1),23(OP2),34(OP3),45(OP4):1DR(1stﾃﾞｨｹｲﾚｰﾄ)     (0～31)
13(OP1),24(OP2),35(OP3),46(OP4):2DR(2ndﾃﾞｨｹｲﾚｰﾄ)     (0～31)
14(OP1),25(OP2),36(OP3),47(OP4):RR(ﾘﾘｰｽﾚｰﾄ)          (0～15)
15(OP1),26(OP2),37(OP3),48(OP4):1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)    (0～15)
16(OP1),27(OP2),38(OP3),49(OP4):TL(ﾄｰﾀﾙﾚﾍﾞﾙ)         (0～127)
17(OP1),28(OP2),39(OP3),50(OP4):KS(ｷｰｽｹｰﾘﾝｸﾞ)        (0～3)
18(OP1),29(OP2),40(OP3),51(OP4):MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)    (0～15)
19(OP1),30(OP2),41(OP3),52(OP4):DT1(ﾃﾞｨﾁｭｰﾝ1)        (0～7)
20(OP1),31(OP2),42(OP3),53(OP4):DT2(ﾃﾞｨﾁｭｰﾝ2)        (0～3)
21(OP1),32(OP2),43(OP3),54(OP4):AME(AMSｲﾈｰﾌﾞﾙ)       (0,1)
```

●音色フォーマット(AL/FB分離方式)

```
00(OP1),11(OP2),22(OP3),33(OP4):AR(ｱﾀｯｸﾚｰﾄ)          (0～31)
01(OP1),12(OP2),23(OP3),34(OP4):1DR(1stﾃﾞｨｹｲﾚｰﾄ)     (0～31)
02(OP1),13(OP2),24(OP3),35(OP4):2DR(2ndﾃﾞｨｹｲﾚｰﾄ)     (0～31)
03(OP1),14(OP2),25(OP3),36(OP4):RR(ﾘﾘｰｽﾚｰﾄ)          (0～15)
04(OP1),15(OP2),26(OP3),37(OP4):1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)    (0～15)
05(OP1),16(OP2),27(OP3),38(OP4):TL(ﾄｰﾀﾙﾚﾍﾞﾙ)         (0～127)
06(OP1),17(OP2),28(OP3),39(OP4):KS(ｷｰｽｹｰﾘﾝｸﾞ)        (0～3)
07(OP1),18(OP2),29(OP3),40(OP4):MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)    (0～15)
08(OP1),19(OP2),30(OP3),41(OP4):DT1(ﾃﾞｨﾁｭｰﾝ1)        (0～7)
09(OP1),20(OP2),31(OP3),42(OP4):DT2(ﾃﾞｨﾁｭｰﾝ2)        (0～3)
10(OP1),21(OP2),32(OP3),43(OP4):AME(AMSｲﾈｰﾌﾞﾙ)       (0,1)
44:AL(ｱﾙｺﾞﾘｽﾞﾑ)          (0～7)
45:FB(ﾌｨｰﾄﾞﾊﾞｯｸ)         (0～7)
46:OM(ｽﾛｯﾄﾏｽｸ)          (0～15)
47:PAN                   (0～3)
48:WF(ｳｪｰｳﾞﾌｫｰﾑ)         (0～3)
49:SYC(ｼﾝｸﾛ)             (0,1)
50:SPD(ｽﾋﾟｰﾄﾞ)           (0～255)
51:PMD                   (0～127)
52:AMD                   (0～127)
53:PMS                   (0～7)
54:AMS                   (0～3)
```

ファンクション$04　ZM_VSET

機能　FM音源音色を登録する
引数　d1.l:音色番号(0-32767)
　　　　d2.l:モード(0=OPMDRV.X互換,1=AL/FB分離方式)
　　　　a1.l:音色データ格納バッファ
戻り値　0:正常終了
　　　　0以外:エラーコード
備考　音色フォーマットについてはZM_VGETとまったく同じ

ファンクション$06　ZM_TEMPO

機能　テンポを設定する
引数　d1.lw:テンポの値(1分間の4分音符の数)
　　　　　　-1=テンポ取得のみを行う
　　　　d1.hw:モード(0=Z-MUSICに登録する(通常),
　　　　　　　　-1=単なるテンポの変更でZ-MUSICには登録しない)
戻り値　設定前のテンポとタイマ値が戻ってくる
　　　　d0.hw:タイマ値
　　　　d0.lw:テンポ値
備考　テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。
　　　　　TIMER-A:20～32767
　　　　　TIMER-B:20～5000
　　　　　YM3802:1～32767

ファンクション$07　ZM_SET_TIMER_VALUE

機能　テンポをタイマ値で設定する
引数　d1.lw:タイマの値
　　　　　　-1=タイマ値取得のみを行う
　　　　d1.hw:モード(0=Z-MUSICに登録する(通常),
　　　　　　　　-1=単なるテンポの変更でZ-MUSICには登録しない)
戻り値　設定前のテンポとタイマ値が戻ってくる
　　　　d0.hw:テンポ値
　　　　d0.lw:タイマ値
備考　テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。
　　　　　TIMER-A:0～1023
　　　　　TIMER-B:0～255
　　　　　YM3802:0～8191

ファンクション$08　ZM_PLAY

機能　演奏開始制御
引数　a1.l:演奏を開始したいトラック番号の並びを格納したアドレス
　　　　　ただしa1.l=0とすると全トラック演奏開始の指定とみなす。
戻り値　a0.l:演奏開始を行えなかったトラック番号の並びを格納したアドレス
備考　引数a1.l、戻り値a0.lともに、
　　　　　{ track(.w),track(.w),.....,track(.w),-1(.w)}
　　　　という構成を取る。ただしトラック番号は0-65534の範囲。

ファンクション$09　ZM_PLAY2

機能　共通コマンドからの再演奏を行う
引数　なし
戻り値　なし
備考　現在Z-MUSICの演奏データバッファに存在する演奏データを初めから(共通コマンドの実行も含めて)演奏する。

ファンクション$0a　ZM_PLAY_AGAIN

機能　再演奏
引数　なし
戻り値　なし
備考　前回行われたファンクションZM_PLAYを再び実行する

ファンクション$0b　ZM_PLAY_STATUS

機能　演奏状態の検査
引数　d1.l:チェックモード(0:全チャンネル検査,1:全トラック検査
　　　　　　2:指定チャンネル検査,3:指定トラック検査)
　　　　d2.hw:デバイスタイプ(0=FM音源,1=AD PCM音源,$8000=MIDI1,
　　　　　　$8001=MIDI2,$8002=MIDI3,$8003=MIDI4,-1=カレントMIDI)
　　　　　　　　　　　　　　　　/d1.l=2のときのみ有効
　　　　d2.lw:チャンネル番号(0-15)　　/d1.l=2のときのみ有効
　　　　d2.l:トラック番号(0-65534)　　/d1.l=3のときのみ有効
　　　　a1.l:検査結果格納バッファアドレス
　　　　　(0にすると検査結果を簡略して返す)/d1.l=0,2のときのみ有効
戻り値　引数a1.l=0のとき、あるいはd1.l=2,3のときのみd0.lに以下の値を返す。
　　　　　　d0.l=0:演奏終了

d0.l=-1:演奏中

備考 ・引数d1.l=0(全チャンネル検査モード)のとき戻り値は引数a1.lから示される領域に演奏中のデバイスタイプ,チャンネルが以下のような書式で返される。

{ active_device(.l),active_device(.l),...,-1(.l) }

なお、active_device(.l)は、

active_device(.hw):デバイスタイプ(0=FM音源,1=AD PCM音源,
$8000=MIDI1,$8001=MIDI2,$8002=MIDI3,
$8003=MIDI4,-1=カレントMIDI)

active_device(.lw):チャンネル番号(0-15)

という書式に準ずる。

・引数d1.l=1(全トラック検査モード)のとき戻り値は引数a1.lから示される領域に演奏中のトラック番号が以下のような書式で返される。

{ active_track(.w),active_track(.w),...,-1(.w) }

なお、active_track(.w)は0-65534の値を取る。

・引数d1.l=2,3のときはd2.lで指定したチャンネルやトラック番号の演奏状態を検査する。このときa1.lを指定する必要はない。検査結果はd0.lに返される。

ファンクション$0c ZM_STOP

機能 演奏停止

引数 a1.l:演奏を停止したいトラック番号の並びを格納したアドレス
ただしa1.l=0とすると全トラック演奏停止の指定とみなす。

戻り値 a0.l:演奏停止を行えなかったトラック番号の並びを格納したアドレス

備考 引数のa1.l、戻り値のa0.lともに、

{ track(.w),track(.w),.....,track(.w),-1(.w) }

という構成を取る。ただしトラック番号は0-65534の範囲

ファンクション$0d ZM_CONT

機能 演奏再開

引数 a1.l:演奏を再開したいトラック番号の並びを格納したアドレス
ただしa1.l=0とすると全トラック演奏再開の指定とみなす。

戻り値 a0.l:演奏再開を行えなかったトラック番号の並びを格納したアドレス

備考 引数のa1.l、戻り値のa0.lともに、

{ track(.w),track(.w),.....,track(.w),-1(.w)}

という構成を取る。ただしトラック番号は0-65534の範囲

ファンクション$0e ZM_ATOI

機能 演奏トラックデータの先頭アドレス取得

引数 d1.l:トラック番号(0-65534)

戻り値 a0.l=0:エラー(指定トラックは存在せず)
a0.l=0以外:指定演奏トラックの先頭アドレス

ファンクション$0f ZM_SET_MASTER_CLOCK

機能 拍子,メトロノーム速度,調号,全音符の絶対音長の設定

引数 d1.l:モード(0=音楽演奏側の設定を行う,1=効果音演奏側の設定を行う)
a1.l:拍子,メトロノーム速度,調号,全音符の絶対音長を格納した領域のアドレス。a1.l=0ならば問い合わせのみ

戻り値 引数のa1.l=0としたときのみ、その時点の設定値を格納した領域のアドレスがa0.lに返る。それ以外は戻り値不定。

備考 d1.l=0で通常音楽演奏側の設定、d1.l=1で効果音演奏側の設定となる。

引数のa1.l、戻り値のa0.lの構造は、

$00(a1).b:拍子分子(1-255)
$01(a1).b:拍子分母(1-255)
$02(a1).b:メトロノーム速度(楽的音長(1-255))
$03(a1).b:reserve
$04(a1).b:調号1(0:C,負値:♭の数,正値:#の数)
$05(a1).b:調号2(Major=0,Minor=1)
$06(a1).w:全音符の絶対音長(1-65535)

という書式に準ずる。

ファンクション$10 ZM_PLAY_ZMD

機能 ZMDの演奏

引数 d2.l:ZMDのサイズ(ヘッダ「$1A,'ZmuSiC',$30」の分は含まれていても含まれていなくてもどちらでもよい)。
d2.l=0を設定するとZ-MUSIC本体のバッファへ演奏データを取り込む動作を省略する。
a1.l:ZMDの格納アドレス(ヘッダ「$1A,'ZmuSiC',$30」を含まず)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 d2.l=0とするには、a1.lで表されるアドレスからの領域の存在が演奏中も保証されることを前提とする。

ファンクション$11 ZM_PLAY_ZMD_SE

機能 ZMDを効果音として演奏する

引数 a1.l:ZMDの格納アドレス(ヘッダ「$1A,'ZmuSiC',$30」を含まず)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。

ファンクション$12 ZM_SE_PLAY

機能 効果音ZMDの演奏

引数 a1.l:ZMDの格納アドレス(トラックチャンネル情報テーブルから)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。
トラックチャンネル情報テーブルについてはMEASURE12を参照。

ファンクション$13 ZM_SE_ADPCM1

機能 (AD)PCM効果音の演奏

引数 d1.hwhb:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)
d1.hwlb:音量(0-127,64:標準)
d1.lwhb:再生周波数(0:3.9kHz 1:5.2kHz 2:7.8kHz
3:10.4kHz 4:15.6kHz)
d1.lwlb:パンポット(0-3 / $80+0-127)
d2.l:再生するAD PCMのデータサイズ
d4.hw:効果音優先度(0-255/0がもっとも低い)
d4.lw:再生チャンネル(0-15)
a1.l:(AD)PCMデータアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。

ファンクション$14 ZM_SE_ADPCM2

機能 Z-MUSICに登録してある(AD)PCM効果音の演奏

引数 d1.hwhb:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)
d1.hwlb:音量(0-127,64:標準)
d1.lwhb:再生周波数
(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz)
d1.lwlb:パンポット(0-3 / $80+0-127)
d2.l:PCMデータ番号(d15が1ならば音色番号($8000+0-32767),
0ならばノート番号(0-32767)指定)
d4.hw:効果音優先度(0-255/0が最も低い)
d4.lw:再生チャンネル(0-15)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

ファンクション$15 ZM_INTERCEPT_PLAY

機能 演奏開始制御の遮断制御

引数 d1.l:-1 演奏開始の遮断を行う
0 演奏開始の遮断の解除と演奏開始
1 演奏開始の遮断の解除

戻り値 なし

備考 演奏開始の遮断を設定するとファンクションZM_PLAYなどの演奏開始命令を実行しても演奏が開始されなくなる。
演奏開始の遮断中に演奏開始を実行した場合、ワークtimer_flgの最上位ビットが1に上書きされる

ファンクション$16 ZM_CURRENT_MIDI_IN

機能 カレントMIDI-IN端子の設定

引数 d1.w:MIDIポートの番号(0-3)
-1では問い合わせのみを行う

戻り値　d0.l=以前のカレントMIDI-INのポート番号(0-3)

ファンクション$17　ZM_CURRENT_MIDI_OUT

機能　カレントMIDI-OUT端子の設定
引数　d1.w:MIDIポートの番号(0-3)
　　　-1では問い合わせのみを行う
戻り値　d0.l=以前のカレントMIDI-OUTのポート番号(0-3)

ファンクション$18　ZM_MIDI_TRANSMISSION

機能　MIDIデータメッセージの送信
引数　d1.l:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.l:送信データサイズ
　　　0とした場合はテキスト型MIDIデータ(MDD)とみなして送信
　　　0以外ではバイナリデータとみなして送信
　　a1.l:送信するMIDIデータを格納したデータバッファアドレス
戻り値　d0.l=送信データの8ビット加算合計値
備考　MDDとはZ-MUSIC ver.2.0相当でMIDIファイルの標準としていたフォーマット。$0d,$0aで始まり16進数のテキストでMIDIメッセージを記述したファイル。
　送信データサイズを指定した場合はバイナリのMIDIメッセージとみなして送信される。Z-MUSICが扱えるバイナリのMIDIメッセージにはスタンダードMIDIファイル(FORMAT0)と単なるバイナリデータ列の2種類がありこのデータの種類の判別はZ-MUSIC ver.3.0が自動で行う。

ファンクション$19　ZM_EXCLUSIVE

機能　エクスクルーシブメッセージの送信
引数　d1.l:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.l:送信データサイズ
　　d3.hw:MAKER ID(0-127)
　　d3.lb:MODEL ID(0-127)
　　d3.hb:DEVICE ID(0-127)
　　a1.l:送信するMIDIデータを格納したデータバッファアドレス
戻り値　なし
備考　エクスクルーシブヘッダの$F0とEOXの$F7は自動送信される。
　d3.hb=-1とすると以前設定したものが選択される。最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
　MAKER IDをRoland($41)とした場合はデータ列をローランドエクスクルーシブフォーマットのデータとみなし、コマンドメッセージ(DT2:$12)やチェックサムバイト付きで送信する。
　MAKER IDをYAMAHA($43)とした場合はデータ列をヤマハエクスクルーシブフォーマットのデータとみなし、データ長やチェックサムバイト付きで送信する。

ファンクション$1b　ZM_SET_EOX_WAIT

機能　EOX送信後のウェイトを設定する
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.w:0-65535
戻り値　d0.l=それまでのウェイト値

ファンクション$1c　ZM_MIDI_INP1

機能　MIDIデータの1バイト入力
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.l:入力モード(0:1回受信を試みてすぐ帰還,
　　　　　　1:受信するまで無限ループ)
戻り値　d0.lに入力結果を返す
備考　戻り値のd0.lは以下のような書式をとる
　　第31ビット:読み込みステータス(1=読みこぼし発生,
　　　　　　　　　0=正常読み出し)
　　第15ビット:データの有無(1=データなし,0=データあり)
　　第0-7ビット:読み出しデータ(0-255)

ファンクション$1d　ZM_MIDI_OUT1

機能　MIDIデータの1バイト出力
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.b:送信データ(0-255)
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード

ファンクション$1e　ZM_MIDI_REC

機能　MIDIデータのレコーディング開始
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード

ファンクション$1f　ZM_MIDI_REC_END

機能　MIDIデータのレコーディング終了
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード

ファンクション$20　ZM_GS_RESET

機能　GS音源の初期化
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d3.b:DEVICE ID(0-127,-1:以前の値を使用)
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード

ファンクション$21　ZM_GS_PARTIAL_RESERVE

機能　GS音源のパーシャルリザーブ
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d3.b:DEVICE ID(0-127,-1:以前の値を使用)
　　a1.l:パート1〜パート16までのパーシャルリザーブ値が格納されているアドレス
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード
備考　a1.lからの領域にパーシャルリザーブ値を16パート分格納しておく。また、合計が楽器の最大同時発声数以下にならなければならない。0(a1)=パート1,1(a1)=パート2…,15(a1)=パート16というふうに対応する。なお、パート10はリズムパートに対応する。
　d3.bを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$22　ZM_GS_REVERB

機能　GS音源のリバーブパラメータの設定
引数　d1.w:MIDIポートの番号(0-3)
　　　-1はカレントMIDI-OUTを使用
　　d2.l:リバーブパラメータ・データサイズ(1-7)
　　d3.b:DEVICE ID(0-127,-1:以前の値を使用)
　　a1.l:リバーブパラメータ格納バッファアドレス
戻り値　d0.l=0:正常終了
　　d0.l=0以外:エラーコード
備考　a1.lからの領域にリバーブパラメータを記述しておく。d2.lは7以内で、指定した個数分のパラメータがGS音源へ送信される。
　d3.bを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):REVERB MACRO	(00:Room1,01:Room2,02:Room3, 03:Hall1,04:Hall2,05:Plate, 06:Delay,07:Panning Delay(FB))
1(a1):REVERB CHARACTER	($00〜$07)
2(a1):REVERB PRE-LPF	($00〜$07)
3(a1):REVERB LEVEL	($00〜$7F)
4(a1):REVERB TIME	($00〜$7F)
5(a1):REVERB DELAY FEEDBACK	($00〜$7F)
6(a1):REVERB SEND LEVEL TO CHORUS	($00〜$7F)

(SC-55マニュアルp79参照)

ファンクション$23　ZM_GS_CHORUS

機能　GS音源のコーラスパラメータの設定

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d2.l:コーラスパラメータデータサイズ(1-8)
　d3.b:DEVICE ID(0-127,-1:以前の値を使用)
　a1.l:コーラスパラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　a1.lからの領域にコーラスパラメータを記述しておく。d2.lは8以内で指定した個数分のパラメータがGS音源へ送信される。
　d3.bを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):CHORUS MACRO	(00:Chorus1,01:Chorus2,02:Chorus3,03:Chorus4,04:Feedback Chorus,05:Flanger,06:Short Delay,07:Short Delay(FB))
1(a1):CHORUS PRE-LPF	($00～$07)
2(a1):CHORUS LEVEL	($00～$7F)
3(a1):CHORUS FEEDBACK	($00～$7F)
4(a1):CHORUS DELAY	($00～$7F)
5(a1):CHORUS RATE	($00～$7F)
6(a1):CHORUS DEPTH	($00～$7F)
7(a1):CHORUS SEND LEVEL TO REVERB	($00～$7F)

(SC-55マニュアルp79参照)

ファンクション$24　ZM_GS_PART_PARAMETER

機能　GS音源のパートパラメータの設定

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d2.l:パートパラメータデータサイズ(1-119)
　d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
　d3.hw:PART NUMBER(1-16)
　a1.l:パートパラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　a1.lからの領域にパートパラメータを記述しておく。d2.lは119以内で指定した個数分のパラメータがGS音源へ送信される。
　d3.bを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Rx CHANNEL	(1～16,17=OFF)
1(a1):Rx PITCH BEND	(0～1:OFF/ON)
2(a1):Rx CH PRESSURE(CAf)	(0～1:OFF/ON)
3(a1):Rx PROGRAM CHANGE	(0～1:OFF/ON)
4(a1):Rx CONTROL CHANGE	(0～1:OFF/ON)
5(a1):Rx POLY PRESSURE(PAf)	(0～1:OFF/ON)
6(a1):Rx NOTE MESSAGE	(0～1:OFF/ON)
7(a1):Rx RPN	(0～1:OFF/ON)
8(a1):Rx NRPN	(0～1:OFF/ON)
9(a1):Rx MODURATION	(0～1:OFF/ON)
10(a1):Rx VOLUME	(0～1:OFF/ON)
11(a1):Rx PANPOT	(0～1:OFF/ON)
12(a1):Rx EXPRESSION	(0～1:OFF/ON)
13(a1):Rx HOLD1	(0～1:OFF/ON)
14(a1):Rx PORTAMENTO	(0～1:OFF/ON)
15(a1):Rx SOSTENUTO	(0～1:OFF/ON)
16(a1):Rx SOFT	(0～1:OFF/ON)
17(a1):MONO/POLY MODE	(0～1:MONO/POLY)
18(a1):ASSIGN MODE	(SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
19(a1):USE FOR RHYTHM PART	(0:OFF,1:MAP1,2:MAP2)
20(a1):PITCH KEY SHIFT	($28～$58:－24～＋24[semitone])
21(a1):PITCH OFFSET FINE(upper)	($08～$F8:-12.0～+12.0[Hz])
22(a1):PITCH OFFSET FINE(lower)	〃
23(a1):PART LEVEL	(0～127)
24(a1):VELOCITY SENSE DEPTH	(0～127)
25(a1):VELOCITY SENSE OFFSET	(0～127)
26(a1):PART PANPOT	(0:RANDOM,01(LEFT)～64～127(RIGHT))
27(a1):KEY RANGE LOW	(0～127:C-1～G9)
28(a1):KEY RANGE HIGH	(0～127:C-1～G9)
29(a1):CC1 CONTROL NUMBER	(0～127)
30(a1):CC2 CONTROL NUMBER	(0～127)
31(a1):CHORUS SEND DEPTH	(0～127)
32(a1):REVERB SEND DEPTH	(0～127)
33(a1):Vibrato rate	($0E～$72:－50～＋50)
34(a1):Vibrato depth	($0E～$72:－50～＋50)
35(a1):TVF cutoff freq.	($0E～$50:－50～＋16)
36(a1):TVF resonance	($0E～$72:－50～＋50)
37(a1):TVF&TVA Env.attack	($0E～$72:－50～＋50)
38(a1):TVF&TVA Env.decay	($0E～$72:－50～＋50)
39(a1):TVF&TVA Env.release	($0E～$72:－50～＋50)
40(a1):Vibrato delay	($0E～$72:－50～＋50)
41(a1):SCALE TUNING C	($00～$7F:－64～＋63[cent])
42(a1):SCALE TUNING C#	($00～$7F:－64～＋63[cent])
43(a1):SCALE TUNING D	($00～$7F:－64～＋63[cent])
44(a1):SCALE TUNING D#	($00～$7F:－64～＋63[cent])
45(a1):SCALE TUNING E	($00～$7F:－64～＋63[cent])
46(a1):SCALE TUNING F	($00～$7F:－64～＋63[cent])
47(a1):SCALE TUNING F#	($00～$7F:－64～＋63[cent])
48(a1):SCALE TUNING G	($00～$7F:－64～＋63[cent])
49(a1):SCALE TUNING G#	($00～$7F:－64～＋63[cent])
50(a1):SCALE TUNING A	($00～$7F:－64～＋63[cent])
51(a1):SCALE TUNING A#	($00～$7F:－64～＋63[cent])
52(a1):SCALE TUNING B	($00～$7F:－64～＋63[cent])
53(a1):MOD PITCH CONTROL	($28～$58:－24～＋24[semitone])
54(a1):MOD TVF CUTOFF CONTROL	($00～$7F:－9600～＋9600[CENT])
55(a1):MOD AMPLITUDE CONTROL	($00～$7F:－100.0～＋100.0[%])
56(a1):MOD LFO1 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
57(a1):MOD LFO1 PITCH DEPTH	($00～$7F:0～600[cent])
58(a1):MOD LFO1 TVF DEPTH	($00～$7F:0～2400[cent])
59(a1):MOD LFO1 TVA DEPTH	($00～$7F:0～100.0[%])
60(a1):MOD LFO2 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
61(a1):MOD LFO2 PITCH DEPTH	($00～$7F:0～600[cent])
62(a1):MOD LFO2 TVF DEPTH	($00～$7F:0～2400[cent])
63(a1):MOD LFO2 TVA DEPTH	($00～$7F:0～100.0[%])
64(a1):BEND PITCH CONTROL	($28～$58:－24～＋24[semitone])
65(a1):BEND TVF CUTOFF CONTROL	($00～$7F:－9600～＋9600[CENT])
66(a1):BEND AMPLITUDE CONTROL	($00～$7F:－100.0～＋100.0[%])
67(a1):BEND LFO1 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
68(a1):BEND LFO1 PITCH DEPTH	($00～$7F:0～600[cent])
69(a1):BEND LFO1 TVF DEPTH	($00～$7F:0～2400[cent])
70(a1):BEND LFO1 TVA DEPTH	($00～$7F:0～100.0[%])
71(a1):BEND LFO2 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
72(a1):BEND LFO2 PITCH DEPTH	($00～$7F:0～600[cent])
73(a1):BEND LFO2 TVF DEPTH	($00～$7F:0～2400[cent])
74(a1):BEND LFO2 TVA DEPTH	($00～$7F:0～100.0[%])
75(a1):CAf PITCH CONTROL	($28～$58:－24～＋24[semitone])
76(a1):CAf TVF CUTOFF CONTROL	($00～$7F:－9600～＋9600[CENT])
77(a1):CAf AMPLITUDE CONTROL	($00～$7F:－100.0～＋100.0[%])
78(a1):CAf LFO1 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
79(a1):CAf LFO1 PITCH DEPTH	($00～$7F:0～600[cent])
80(a1):CAf LFO1 TVF DEPTH	($00～$7F:0～2400[cent])
81(a1):CAf LFO1 TVA DEPTH	($00～$7F:0～100.0[%])
82(a1):CAf LFO2 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
83(a1):CAf LFO2 PITCH DEPTH	($00～$7F:0～600[cent])
84(a1):CAf LFO2 TVF DEPTH	($00～$7F:0～2400[cent])
85(a1):CAf LFO2 TVA DEPTH	($00～$7F:0～100.0[%])
86(a1):PAf PITCH CONTROL	($28～$58:－24～＋24[semitone])
87(a1):PAf TVF CUTOFF CONTROL	($00～$7F:－9600～＋9600[CENT])
88(a1):PAf AMPLITUDE CONTROL	($00～$7F:－100.0～＋100.0[%])
89(a1):PAf LFO1 RATE CONTROL	($00～$7F:－10.0～＋10.0[Hz])
90(a1):PAf LFO1 PITCH DEPTH	($00～$7F:0～600[cent])
91(a1):PAf LFO1 TVF DEPTH	($00～$7F:0～2400[cent])
92(a1):PAf LFO1 TVA DEPTH	($00～$7F:0～100.0[%])

```
 93(a1):PAf LFO2 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
 94(a1):PAf LFO2 PITCH DEPTH          ($00～$7F:0～600[cent])
 95(a1):PAf LFO2 TVF DEPTH           ($00～$7F:0～2400[cent])
 96(a1):PAf LFO2 TVA DEPTH            ($00～$7F:0～100.0[%])

 97(a1):CC1 PITCH CONTROL     ($28～$58:－24～＋24[semitone])
 98(a1):CC1 TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
 99(a1):CC1 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
100(a1):CC1 LFO1 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
101(a1):CC1 LFO1 PITCH DEPTH          ($00～$7F:0～600[cent])
102(a1):CC1 LFO1 TVF DEPTH           ($00～$7F:0～2400[cent])
103(a1):CC1 LFO1 TVA DEPTH            ($00～$7F:0～100.0[%])
104(a1):CC1 LFO2 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
105(a1):CC1 LFO2 PITCH DEPTH          ($00～$7F:0～600[cent])
106(a1):CC1 LFO2 TVF DEPTH           ($00～$7F:0～2400[cent])
107(a1):CC1 LFO2 TVA DEPTH            ($00～$7F:0～100.0[%])

108(a1):CC2 PITCH CONTROL     ($28～$58:－24～＋24[semitone])
109(a1):CC2 TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
110(a1):CC2 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
111(a1):CC2 LFO1 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
112(a1):CC2 LFO1 PITCH DEPTH          ($00～$7F:0～600[cent])
113(a1):CC2 LFO1 TVF DEPTH           ($00～$7F:0～2400[cent])
114(a1):CC2 LFO1 TVA DEPTH            ($00～$7F:0～100.0[%])
115(a1):CC2 LFO2 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
116(a1):CC2 LFO2 PITCH DEPTH          ($00～$7F:0～600[cent])
117(a1):CC2 LFO2 TVF DEPTH           ($00～$7F:0～2400[cent])
118(a1):CC2 LFO2 TVA DEPTH            ($00～$7F:0～100.0[%])
```
(SC-55のマニュアルp75,p79～p80参照)

ファンクション$25 ZM_GS_DRUM_PARAMETER(ZM_GS_DRUM_SETUP)

機能 GS音源のドラムセットアップ設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ドラムセットアップパラメータデータサイズ(1-8)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hwlb:NOTE NUMBER(0-127)
d2.hwhb:MAP NUMBER(0-1)
a1.l:ドラムセットアップパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にドラムセットアップパラメータを記述しておく。
d2.lは８以内で指定した個数分のパラメータがGS音源へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
0(a1):PLAY KEY NUMBER          (0～127)
1(a1):LEVEL                    (0～127)
2(a1):ASSIGN GROUP NUMBER      (0:NON,1～127)
3(a1):PANPOT          (0:RANDOM,01(LEFT)～64～127(RIGHT))
4(a1):REVERB DEPTH             (0～127:0.0～1.0)
5(a1):CHORUS DEPTH             (0～127:0.0～1.0)
6(a1):Rx NOTE OFF              (0～1:OFF/ON)
7(a1):Rx NOTE ON               (0～1:OFF/ON)
```
(SC-55マニュアルp75,p82参照)

ファンクション$26 ZM_GS_DRUM_NAME

機能 GS音源のドラムセット名設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ドラムセットアップパラメータデータサイズ(1-8)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:MAP NUMBER(0,1)
a1.l:ドラムセット名文字列格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 文字列は12文字以内でa1.lからの領域に文字列を記述しておく。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
(SC-55マニュアルp82参照)

ファンクション$27 ZM_GS_PRINT

機能 GS音源の画面へのメッセージ出力

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:メッセージ長(1-32)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:文字列格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 メッセージは32文字以内でa1.lからの領域に文字列を記述しておく。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$28 ZM_GS_DISPLAY

機能 GS音源の画面へのドットパターン出力

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:ドットパターン格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にドットパターンを記述しておく。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

例
```
  /*SC-55の画面に"善"を表示する*/
        moveq.l #-1,d1
        moveq.l #-1,d3
        lea     data(pc),a1
        Z_MUSIC #ZM_GS_DISPLAY
        rts
data:
        dc.w    %0001000000010000
        dc.w    %0000100000100000
        dc.w    %0111111111111100
        dc.w    %0000000100000000
        dc.w    %0011111111111000
        dc.w    %0000000100000000
        dc.w    %0111111111111100
        dc.w    %0001000100010000
        dc.w    %0000100100100000
        dc.w    %1111111111111110
        dc.w    %0000000000000000
        dc.w    %0011111111111000
        dc.w    %0010000000001000
        dc.w    %0010000000001000
        dc.w    %0011111111111000
        dc.w    %0010000000001000
```

ファンクション$29 ZM_GM_SYSTEM_ON

機能 GM音源のリセット

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

ファンクション$30 ZM_MT32_RESET

機能 MT-32の初期化

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$31 ZM_MT32_PARTIAL_RESERVE

機能 MT-32のパーシャルリザーブ

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:パート1～パート9までのパーシャルリザーブ値が
　格納されているバッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にパーシャルリザーブ値を9パート分格納しておく。また、合計が楽器の最大同時発声数以下にならなければならない。0(a1)=パート1,1(a1)=パート2…,8(a1)=パート9というふうに対応する。なお8(a1)のパート9はリズムパートに対応する。d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$32 ZM_MT32_REVERB

機能 MT-32のリバーブパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:リバーブパラメータデータサイズ(1-3)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:リバーブパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にリバーブパラメータを記述しておく。d2.lは3以内で、指定した個数分のパラメータがMT-32へ送信される。d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

0(a1):REVERB MODE	(0:Room,1:Hall,2:Plate,3:Tap Delay)
1(a1):REVERB TIME	(0～7)
2(a1):REVERB LEVEL	(0～7)

(MT-32マニュアルp35,CM-64マニュアルp30参照)

ファンクション$33 ZM_MT32_PART_SETUP

機能 MT-32のパートパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:パートパラメータデータサイズ(1-9)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:パートパラメータ格納バッファ

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にパートパラメータを記述しておく。
パートパラメータは,
0(a1)=パート1のMIDIチャンネル,
1(a1)=パート2のMIDIチャンネル,
　:
8(a1)=リズムパートのMIDIチャンネル
という構成を取る
d2.lは9以内で指定した個数分のパラメータがMT-32へ送信される。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$34 ZM_MT32_DRUM

機能 MT-32のドラムセットアップ設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:ドラムセットアップパラメータデータサイズ(1-4)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:NOTE NUMBER(24～87)
a1.l:ドラムセットアップパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にドラムセットアップパラメータを記述しておく。
d2.lは4以内で指定した個数分のパラメータがMT-32へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):TIMBRE	(0～63:i1～64,64～127:r1～64)
1(a1):OUTPUT LEVEL	(0～100)
2(a1):PANPOT	(0～14)
3(a1):REVERB SWITCH	(0～1:OFF/ON)

(MT-32マニュアルp35,CM-64マニュアルp30参照)

ファンクション$35 ZM_MT32_COMMON

機能 MT-32のコモンパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:コモンパラメータデータサイズ(1-4)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:PROGRAM NUMBER(1-64)
a1.l:コモンパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にコモンパラメータを記述しておく。
d2.lは4以内で指定した個数分のパラメータがMT-32へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Structure of Partial#1&2	(0～12:1～13)
1(a1):Structure of Partial#3&4	(0～12:1～13)
2(a1):PARTIAL MUTE	(0～15)
3(a1):ENV MODE	(0～1:NORMAL,NO SUSTAIN)

(MT-32マニュアルp34,CM-64マニュアルp29参照)

ファンクション$36 ZM_MT32_PARTIAL

機能 MT-32のパーシャルパラメータ設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:パーシャルパラメータデータサイズ(1-58)
d3.lw:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d3.hwlb:PARTIAL NUMBER(1-4)
d3.hwhb:PROGRAM NUMBER(1-64)
a1.l:パーシャルパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にパーシャルパラメータを記述しておく。
d2.lは58以内で指定した個数分のパラメータがMT-32へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):WG PITCH COARSE	(0～96:C1,C#1,～,C9)
1(a1):WG PITCH FINE	(0～100:-50～+50)
2(a1):WG PITCH KEYFOLLOW	(0～16:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2,s1,s2)
3(a1):WG PITCH BENDER SW	(0～1:OFF/ON)
4(a1):WG WAVEFORM/PCM BANK	(0～3:SQU1,SAW1,SQU2,SAW2)
5(a1):WG PCM WAVE	(0～127:1～128)
6(a1):WG PULSE WIDTH	(0～100)
7(a1):WG PW VELO SENS	(0～14:－7～＋7)
8(a1):P-ENV DEPTH	(0～10)
9(a1):P-ENV VELO SENS	(0～3)
10(a1):P-ENV TIME KEYF	(0～4)
11(a1):P-ENV TIME 1	(0～100)
12(a1):P-ENV TIME 2	(0～100)
13(a1):P-ENV TIME 3	(0～100)
14(a1):P-ENV TIME 4	(0～100)
15(a1):P-ENV LEVEL 0	(0～100:-50～+50)
16(a1):P-ENV LEVEL 1	(0～100:-50～+50)
17(a1):P-ENV LEVEL 2	(0～100:-50～+50)
18(a1):P-ENV SUSTAIN LEVEL	(0～100:-50～+50)
19(a1):END LEVEL	(0～100:-50～+50)
20(a1):P-LFO RATE	(0～100)
21(a1):P-LFO DEPTH	(0～100)
22(a1):P-LFO MOD SENS	(0～100)
23(a1):TVF CUTOFF FREQ	(0～100)
24(a1):TVF RESONANCE	(0～30)

25(a1):TVF KEYFOLLOW (0～14:－1,－1/2,－1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2)
26(a1):TVF BIAS POINT/DIR (0～127)
27(a1):TVF BIAS LEVEL (0～14:－7～＋7)
28(a1):TVF ENV DEPTH (0～100)
29(a1):TVF ENV VELO SENS (0～100)
30(a1):TVF ENV DEPTH KEYF (0～4)
31(a1):TVF ENV TIME KEYF (0～4)
32(a1):TVF ENV TIME 1 (0～100)
33(a1):TVF ENV TIME 2 (0～100)
34(a1):TVF ENV TIME 3 (0～100)
35(a1):TVF ENV TIME 4 (0～100)
36(a1):TVF ENV TIME 5 (0～100)
37(a1):TVF ENV LEVEL 1 (0～100)
38(a1):TVF ENV LEVEL 2 (0～100)
39(a1):TVF ENV LEVEL 3 (0～100)
40(a1):TVF ENV SUSTAIN LEVEL (0～100)
41(a1):TVA LEVEL (0～100)
42(a1):TVA VELO SENS (0～100:-50～+50)
43(a1):TVA BIAS POINT 1 (0～127)
44(a1):TVA BIAS LEVEL 1 (0～12:-12～0)
45(a1):TVA BIAS POINT 2 (0～127)
46(a1):TVA BIAS LEVEL 2 (0～12:-12～0)
47(a1):TVA ENV TIME KEYF (0～4)
48(a1):TVA ENV TIME V_FOLLOW (0～4)
49(a1):TVA ENV TIME 1 (0～100)
50(a1):TVA ENV TIME 2 (0～100)
51(a1):TVA ENV TIME 3 (0～100)
52(a1):TVA ENV TIME 4 (0～100)
53(a1):TVA ENV TIME 5 (0～100)
54(a1):TVA ENV LEVEL 1 (0～100)
55(a1):TVA ENV LEVEL 2 (0～100)
56(a1):TVA ENV LEVEL 3 (0～100)
57(a1):TVA ENV SUSTAIN LEVEL (0～100)
(MT-32マニュアルp34,CM-64マニュアルp29参照)

ファンクション$37 ZM_MT32_PATCH

機能 MT-32のパッチパラメータ設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:パッチパラメータデータサイズ(1-7)
d3.lw:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:PATCH NUMBER(1-128)
a1.l:パッチパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にパッチパラメータを記述しておく。
d2.lは7以内で指定した個数分のパラメータがMT-32へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):TIMBRE GROUP (0～3:a,b,i,r)
1(a1):TIMBRE NUMBER (0～63)
2(a1):KEY SHIFT (0～48:－24～＋24)
3(a1):FINE TUNE (0～100:－50～＋50)
4(a1):BENDER RANGE (0～24)
5(a1):ASSIGN MODE (0～3:POLY1～4)
6(a1):REVERB SWITCH (0～1:OFF/ON)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

ファンクション$38 ZM_MT32_PRINT

機能 MT-32の画面へのメッセージ出力

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:メッセージ長(1-20)
d3.w:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
a1.l:文字列格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 メッセージは20文字以内でa1.lからの領域に文字列を記述しておく。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$39 ZM_U220_SETUP

機能 U220のセットアップパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:セットアップパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にセットアップパラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Chorus SW (0～1:OFF/ON)
1(a1):Reverb SW (0～1:OFF/ON)
2(a1):Rx Cntrl Channel (1～17:1～16,OFF)
3(a1):Patch Change (0～5:MAP1～MAP4,Dir,OFF)
4(a1):Timbre Change (0～5:MAP1～MAP4,Dir,OFF)
5(a1):Rhythm Change (0～5:MAP1～MAP4,Dir,OFF)
6(a1):R.Inst Assign (0～5:MAP1～MAP4,Dir,OFF)
(U220マニュアルp46,p146～148参照)

注意 U220用の命令はU20に対しても使用できる。以下同様。

ファンクション$3a ZM_U220_PART_SETUP

機能 U220の(テンポラリパッチ)パートセットアップパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:PART NUMBER(1-6)
a1.l:パートセットアップパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にパートセットアップパラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Timbre Number (1～128)
1(a1):Voice Reserve (0～30)
2(a1):Receive Channel (1～17:1～16,OFF)
3(a1):Key Range Low (0～127:C-1～G9)
4(a1):Key Range High (0～127:C-1～G9)
5(a1):Velo Level (0～1:Above/Below)
6(a1):Velo Threshold (0～127)
7(a1):Output Assign (0～4:Dry,Rev,Cho,Dir1,Dir2)
8(a1):Level (0～127)
9(a1):Pan (0～15:L7～M～R7)
10(a1):Rx Volume (0～1:OFF/ON)
11(a1):Rx Pan (0～1:OFF/ON)
12(a1):Rx Hold (0～1:OFF/ON)
(U220マニュアルp58,p149参照)

ファンクション$3b ZM_U220_COMMON

機能 U220の(テンポラリパッチ)コモンパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:コモンパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にコモンパラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Chorus Type (0～4:Chorus1,Chorus2,FB-Chorus,

	Flanger,Short Delay)
1(a1):Chorus Out Mode	(0〜1:Pre Rev,Post Rev)
2(a1):Chorus Level	(0〜31)
3(a1):Chorus Delay	(0〜31)
4(a1):Chorus Rate	(0〜31)
5(a1):Chorus Depth	(0〜31)
6(a1):Chorus Feedback	(1〜63:−31〜0〜+31)
7(a1):Reverb Type	(0〜7:Room1〜3,Hall1〜2,Gate Delay,CrossDelay)
8(a1):Reverb Time	(0〜31)
9(a1):Reverb Level	(0〜31)
10(a1):Reverb Delay Feedback	(0〜31)
11(a1):Reverb Pre Delay Feedback	(0〜31)
12(a1):Param1 #	(0〜63:0〜5,7〜31,64〜95,OFF)
13(a1):Param1 Param	(0:Timbre Level,1:Env Attack 2:Env Decay,3:Env Sustain 4:Env Release,5:A.Bend Depth 6:A.Bend Rate,7:Detune Depth 8:Vib Rate,9:Vib Wave Form 10:Vib Depth,11:Vib Delay 12:VibRiseTime,13:VibModDepth 14:ChorusLevel,15:Chorus Rate 16:Chorus Feedback, 17:ReverbLevel,18:DelayFeedback)
14(a1):Param2 #	(Same as Param1 #)
15(a1):Param2 Param	(Same as Param1 Param)
16(a1):Param3 #	(Same as Param1 #)
17(a1):Param3 Param	(Same as Param1 Param)

(U220マニュアルp54,p149参照)

ファンクション$3c　ZM_U220_TIMBRE

機能　U220のティンバーパラメータの設定

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d3.lw:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
　d3.hw:PROGRAM NUMBER(1-128)
　a1.l:ティンバーパラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　a1.lからの領域にティンバーパラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Tone Media	(0〜31:I,1〜31)
1(a1):Tone Number	(1〜128)
2(a1):Timbre Level	(0〜127)
3(a1):Velocity Sens	(1〜15:−7〜+7)
4(a1):Channel Press Sens	(1〜15:−7〜+7)
5(a1):Env Attack Rate	(1〜15:−7〜+7)
6(a1):Env Decay Rate	(1〜15:−7〜+7)
7(a1):Env Sustain Level	(1〜15:−7〜+7)
8(a1):Env Release Rate	(1〜15:−7〜+7)
9(a1):Pitch Shift Coarse	(8〜56:−24〜+24)
10(a1):Pitch Shift Fine	(14〜114:−50〜+50)
11(a1):Bend Range Lower	(0〜15:−36,−24,−12〜0)
12(a1):Bend Range Upper	(0〜12)
13(a1):Channel After Sens	(0〜27:−36,−24,−12〜+12)
14(a1):Poly After Sens	(0〜27:−36,−24,−12〜+12)
15(a1):Auto Bend Depth	(0〜27:−36,−24,−12〜+12)
16(a1):Auto Bend Rate	(0〜15)
17(a1):Detune Depth	(0〜15)
18(a1):Rate	(0〜63)
19(a1):Waveform	(0〜8)
20(a1):Depth	(0〜15)
21(a1):Delay	(0〜15)
22(a1):Rise Time	(0〜15)
23(a1):Modulation Depth	(0〜15)
24(a1):Ch After Sens	(0〜15)
25(a1):Poly After Sens	(0〜15)

(U220マニュアルp65,p149参照)

ファンクション$3d　ZM_U220_DRUM_SETUP

機能　U220の(テンポラリパッチ)パッチドラムセットアップ設定

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d3.w:DEVICE ID(0-127,-1:以前の値を使用)
　a1.l:ドラムセットアップパラメータ格納バッファ

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　a1.lからの領域にドラムセットアップパラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Rhythm Setup #	(0〜3:1〜4)
1(a1):Voice Reserve	(0〜30)
2(a1):Receive Channel	(1〜17:1〜16,OFF)
3(a1):Level	(0〜127)
4(a1):Level Boost Sw	(0〜1:OFF/ON)
5(a1):Rx Volume	(0〜1:OFF/ON)
6(a1):Rx Hold	(0〜1:OFF/ON)

(U220マニュアルp63,p149参照)

ファンクション$3e　ZM_U220_DRUM_INST

機能　U220の(テンポラリパッチ)ドラム音色設定

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
　d3.hw:NOTE NUMBER(35-99)
　a1.l:ドラム音色パラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　a1.lからの領域にドラム音色パラメータを記述しておく。
a1.lの構成は必ず下に示す書式に従う。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):Tone Media	(0〜31:I,1〜31)
1(a1):Tone Number	(1〜128)
2(a1):Source Key	(0〜127:C−1〜G9)
3(a1):Mute Inst	(34〜98:OFF,B1〜D7)
4(a1):Inst Level	(0〜31)
5(a1):Velocity Sens	(0〜15)
6(a1):Env Mode	(0〜1:Sustain,No Sustain)
7(a1):Env Attack Rate	(1〜15:−7〜+7)
8(a1):Env Decay Rate	(1〜15:−7〜+7)
9(a1):Env Release Rate	(1〜15:−7〜+7)
10(a1):Pitch Shift Coarse	(0〜27:-36,-24,-12〜+12)
11(a1):Pitch Shift Fine	(14〜114:−50〜+50)
12(a1):Channel After Sens	(0〜27:-36,-24,-12〜+12)
13(a1):Poly After Sens	(0〜27:-36,-24,-12〜+12)
14(a1):Random	(0〜15)
15(a1):Auto Bend Depth	(0〜27:-36,-24,-12〜+12)
16(a1):Auto Bend Rate	(0〜15)
17(a1):Detune Depth	(0〜15)
18(a1):Output Assign	(0〜3:Dry,Rev,Cho,Dir1)
19(a1):Pan	(0〜15:L7〜M〜R7)

(U220マニュアルp72,p149)

ファンクション$3f　ZM_U220_PRINT

機能　U220の画面へのメッセージ出力

引数　d1.w:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
　d2.l:メッセージ長(1-12)
　d3.w:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
　a1.l:文字列格納バッファアドレス

戻り値　d0.l=0:正常終了
　d0.l=0以外:エラーコード

備考　メッセージは12文字以内でa1.lからの領域に文字列を記述して

おく。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$40 ZM_M1_SETUP

機能 M1のSEQ-SONG0のMIDIチャンネルの設定
引数 a1.l:8パート分のMIDIチャンネル列格納バッファアドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lで表される領域には必ず8パート分のMIDIチャンネルを記載しておく。

0(a1):パート1のMIDIチャンネル
1(a1):パート2のMIDIチャンネル
2(a1):パート3のMIDIチャンネル
3(a1):パート4のMIDIチャンネル
4(a1):パート5のMIDIチャンネル
5(a1):パート6のMIDIチャンネル
6(a1):パート7のMIDIチャンネル
7(a1):パート8のMIDIチャンネル

注意 M1用の命令はM1EX/M1R/M1REXに対しても使用できる。以下同様。

ファンクション$41 ZM_M1_PART_SETUP

機能 M1のSEQ-SONG0のトラックパラメータの設定
引数 a1.l:8パート分のトラックパラメータ格納バッファアドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lで表される領域には必ず8パート分のトラックパラメータを記載しておく。

```
TRACK 1
0(a1):PROGRAM NUMBER        (0～199:100～C99)
1(a1):OUTPUT LEVEL          (0～99)
2(a1):KEY TRANSPOSE         ($F4～$0C:－12～12)
3(a1):DETUNE                ($CE～$32:－50～＋50)
4(a1):PAN                   (0～13:10:0～0:10,C,C+D,D)
TRACK 2～8
以下同様
(M1マニュアルp126参照)
```

ファンクション$42 ZM_M1_EFFECT_SETUP

機能 M1のSEQ-SONG0のエフェクトパラメータの設定
引数 a1.l:エフェクトパラメータ格納バッファアドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lで表される領域にはダミーデータも含めて必ず25個のエフェクトパラメータが記載されていなければならない。

```
0(a1):Effect 1 Pattern No.      (0～33:1～32,Thru)
1(a1):Effect 2 Pattern No.      (0～33:1～32,Thru)
2(a1):Effect 1 L-Ch E.Balnc     (0～100)
3(a1):Effect 1 R-Ch E.Balnc     (0～100)
4(a1):Effect 2 L-Ch E.Balnc     (0～100)
5(a1):Effect 2 R-Ch E.Balnc     (0～100)
6(a1):Output 3 Pan         (0～101:OFF,100:0～0:100)
7(a1):Output 4 Pan         (0～101:OFF,100:0～0:100)
8(a1):Effect I/O    (BIT0:Effect 1 L-Ch 0=OFF/1=ON
                     BIT1:Effect 1 R-Ch 0=OFF/1=ON
                     BIT2:Effect 2 L-Ch 0=OFF/1=ON
                     BIT3:Effect 2 R-Ch 0=OFF/1=ON)
                     BIT4:Effect 2 0=PARALLEL/1=SERIAL)
effect[9～16]:Effect 1 Parameter
effect[17～24]:Effect 2 Parameter
```

Structure of Effect Parameter

(ofs=9 or 17)

```
1～3:Hall (4,5:Room,6:Live Stage)
ofs+0(a1):Reverb Time     (0～97:0.2～9.9),(0～48:0.2～5.0)
ofs+1(a1):Dummy                   0
ofs+2(a1):High Damp               (0～99)
ofs+3(a1):Pre Delay               (0～200)
ofs+4(a1):E/R Level               (0～99)
ofs+5(a1):Dummy                   0
ofs+6(a1):EQ High                 ($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low                  ($F4～$0C:－12～＋12)

7～9:Early Reflection 1,2,3
ofs+0(a1):E/R Time                (0～70:100～800)
ofs+1(a1):Pre Delay               (0～200)
ofs+2(a1):Dummy                   0
    :                             :
ofs+5(a1):Dummy                   0
ofs+6(a1):EQ High                 ($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low                  ($F4～$0C:－12～＋12)

10:Stereo Delayt,11:Cross Delay
ofs+0(a1):Delay Time L (L)        (0～500)
ofs+1(a1):Delay Time L (H)
ofs+2(a1):Feedback                ($9D～$63:－99～99)
ofs+3(a1):High Damp               (0～99)
ofs+4(a1):Delay Time R (L)        (0～500)
ofs+5(a1):Delay Time R (H)
ofs+6(a1):EQ High                 ($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low                  ($F4～$0C:－12～＋12)

12～13:Stereo Chorus 1～2 (14～15:Flanger)
ofs+0(a1):Depth                   (0～99)
ofs+1(a1):Speed             (0～99:0.03～3.00 0.03step
                             100～199:3.1～13.0 0.1step
                             200～216:14.0～30.0 1.0step)
ofs+2(a1):LFO Status       (BIT0:Waveform =0:Sin,=1:Tri
                             BIT1:Phase =0:0.0°,=1:180°
                             BIT2:WaveShape=0:Normal
                                           =1:for Flanger)
ofs+3(a1):Feedback                ($9D～$63:－99～＋99)
ofs+4(a1):Delay Time              (0～200),(0～50)
ofs+5(a1):Dummy                   0
ofs+6(a1):EQ High                 ($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low                  ($F4～$0C:－12～＋12)

16～17:Phase Shifter 1～2
ofs+0(a1):Depth                   (0～99)
ofs+1(a1):Speed             (0～99:0.03～3.00 0.03step
                             100～199:3.1～13.0 0.1step
                             200～216:14.0～30.0 1.0step)
ofs+2(a1):LFO Status       (BIT0:Waveform =0:Sin,=1:Tri
                             BIT1:Phase =0:0.0°,=1:180°
                             BIT2:Wave Shape=0:Normal
                                            =1:for Flanger)
ofs+3(a1):Feedback                ($9D～$63:－99～＋99)
ofs+4(a1):Manual                  (0～99)
ofs+5(a1):Dummy                   0
ofs+6(a1):Dummy                   0
ofs+7(a1):Dummy                   0

18～19:Stereo Toremolo 1～2
ofs+0(a1):Depth                   (0～99)
ofs+1(a1):Speed             (0～99:0.03～3.00 0.03step
                             100～199:3.1～13.0 0.1step
                             200～216:14.0～30.0 1.0step)
ofs+2(a1):LFO Status       (BIT0:Waveform =0:Sin,=1:Tri
                             BIT1:Phase =0:0.0°,=1:180°
                             BIT2:Wave Shape=0:Normal
                                            =1:for Flanger)
ofs+3(a1):Shape                   ($9D～$63:－99～＋99)
ofs+4(a1):Dummy                   0
ofs+5(a1):Dummy                   0
ofs+6(a1):EQ High                 ($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low                  ($F4～$0C:－12～＋12)
```

20:Equalizer

ofs+0(a1):Dummy	0
:	:
ofs+3(a1):Dummy	0
ofs+4(a1):Low fc	(0～2:0.25k,0.50k,1.00k)
ofs+5(a1):High fc	(0～2:1k,2k,4k)
ofs+6(a1):High Gain	($F4～$0C:－12～＋12)
ofs+7(a1):Low Gain	($F4～$0C:－12～＋12)

21:Overdrive

ofs+0(a1):Dummy	0
ofs+1(a1):Dummy	0
ofs+2(a1):Drive	(0～99)
ofs+3(a1):Level	(0～99)
ofs+4(a1):Dummy	0
ofs+5(a1):Dummy	0
ofs+6(a1):EQ High	($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low	($F4～$0C:－12～＋12)

22:Distortion

ofs+0(a1):Dummy	0
ofs+1(a1):Dummy	0
ofs+2(a1):Distortion	(0～99)
ofs+3(a1):Level	(0～99)
ofs+4(a1):Dummy	0
ofs+5(a1):Dummy	0
ofs+6(a1):Dummy	0
ofs+7(a1):EQ Low Gain	($F4～$0C:－12～＋12)

23:Exciter

ofs+0(a1):Blend	($9D～$63:－99～＋99)
ofs+1(a1):Emphatic Point	(0～9:1～10)
ofs+2(a1):Dummy	0
:	:
ofs+5(a1):Dummy	0
ofs+6(a1):EQ High	($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low	($F4～$0C:－12～＋12)

24:Synphonic Ensamble

ofs+0(a1):Depth	(0～99)
ofs+1(a1):Dummy	0
:	:
ofs+5(a1):Dummy	0
ofs+6(a1):EQ High	($F4～$0C:－12～＋12)
ofs+7(a1):EQ Low	($F4～$0C:－12～＋12)

25:Roteffect Speaker

ofs+0(a1):Depth	(0～99)
ofs+1(a1):Dummy	0
ofs+2(a1):Speed Rate	($F6～$0A:－10～＋10)
ofs+3(a1):Dummy	0
:	:
ofs+7(a1):Dummy	0

26:Delay/Hall

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):Reverb Time	(0～97:0.2～9.9)
ofs+5(a1):Dummy	0
ofs+6(a1):High Damp	(0～99)
ofs+7(a1):Pre Delay	(0～150)

27:Delay/Room

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):Reverb Time	(0～97:0.2～9.9)
ofs+5(a1):Dummy	0
ofs+6(a1):High Damp	(0～99)
ofs+7(a1):Pre Delay	(0～150)

28:Delay/Early Reflection

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):E/R Time	(0～30:100～400)
ofs+5(a1):Pre Delay	(0～150)
ofs+6(a1):Dummy	0
ofs+7(a1):Dummy	0

29:Delay/Chorus

ofs+0(a1):Delay Time L (L)	(0～500)
ofs+1(a1):Delay Time L (H)	
ofs+2(a1):Feedback L	($9D～$63:－99～99)
ofs+3(a1):High Damp L	(0～99)
ofs+4(a1):Delay Time R (L)	(0～500)
ofs+5(a1):Delay Time R (H)	
ofs+6(a1):Feedback R	($9D～$63:－99～99)
ofs+7(a1):High Damp R	(0～99)

30:Delay/Chorus (31:Delay Flanger)

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):Depth	(0～99)
ofs+5(a1):Speed	(0～99:0.03～3.00 0.03step 100～199:3.1～13.0 0.1step 200～216:14.0～30.0 1.0step)
ofs+6(a1):LFO Status	(BIT0:Waveform =0:Sin,=1:Tri BIT1:Phase =0:0.0°,=1:180° BIT2:Wave Shape=0:Normal =1:for Flanger)
ofs+7(a1):Feedback	0,($9D～$63:-99～99)

32:Delay/Phaser

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):Depth	(0～99)
ofs+5(a1):Speed	(0～99:0.03～3.00 0.03step)
ofs+6(a1):Feedback	($9D～$63:－99～99)
ofs+7(a1):Dummy	0

33:Delay/Tremolo

ofs+0(a1):Delay Time (L)	(0～500)
ofs+1(a1):Delay Time (H)	
ofs+2(a1):Feedback	($9D～$63:－99～99)
ofs+3(a1):High Damp	(0～99)
ofs+4(a1):Depth	(0～99)
ofs+5(a1):Speed	(0～99:0.03～3.00 0.03step)
ofs+6(a1):Dummy	0
ofs+7(a1):Shape	($9D～$63:－99～99)

(M1マニュアルp127参照)

ファンクション$43 ZM_M1_PRINT

機能	M1のSEQ-SONG0のタイトル設定
引数	d2.l:メッセージ長(1-10) a1.l:文字列格納バッファアドレス
戻り値	d0.l=0:正常終了 d0.l=0以外:エラーコード
備考	メッセージは10文字以内でa1.lからの領域に文字列を記述しておく。

ファンクション$44 ZM_SEND_TO_M1

機能 M1へパラメータを送信する

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d3.w:DEVICE ID(0-127,-1:以前の値を使用)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 DEVICE ID=&H30+(グローバルチャンネル-1)
グローバルチャンネルはGLOBALモードのF5-1で設定できる。
d3.wは省略可能。省略すると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$30が選択される。
必ずファンクションM1_SETUP～ファンクションM1_PRINTを設定してからこのファンクションを実行すること。
ファンクションM1_EFFECT_SETUPのみ省略が可能。このろきはZ-MUSICが持つデフォルトデータが送信される。
ファンクションM1_SETUP～ファンクションM1_PRINTの間にM1関係以外のファンクションコールがあると正常なデータが送信されない。

例

```
        lea     midi(pc),a1
        Z_MUSIC #M1_SETUP

        lea     setup(pc),a1
        Z_MUSIC #M1_PART_SETUP

        lea     print(pc),a1
        Z_MUSIC #M1_PRINT

        moveq.l #-1,d1
        moveq.l #-1,d3
        Z_MUSIC #SEND_TO_M1
        rts

midi:   dc.b    2,1,4,3,6,5,8,7
setup:  /*音色,音量,ｷｰﾄﾗﾝｽﾎﾟｰｽﾞ,ﾃﾞﾁｭｰﾝ,ﾊﾟﾝのﾌｫｰﾏｯﾄ*/
        dc.b    0,99,0,0,5      *TRACK 1
        dc.b    0,99,0,0,5      *TRACK 2
        dc.b    0,99,0,0,5      *TRACK 3
        dc.b    0,99,0,0,5      *TRACK 4
        dc.b    0,99,0,0,5      *TRACK 5
        dc.b    0,99,0,0,5      *TRACK 6
        dc.b    0,99,0,0,5      *TRACK 7
        dc.b    0,99,0,0,5      *TRACK 8
print:  dc.b    'Z.Nishikawa',0
```

ファンクション$46 ZM_SC88_MODE_SET

機能 SC-88モード設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:SC-88 MODE(0:Single, 1:Double)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

ファンクション$47 ZM_SC88_REVERB

機能 SC-88のリバーブパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:リバーブパラメータデータサイズ(1-8)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:リバーブパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にリバーブパラメータを記述しておく。d2.lは8以内で、指定した個数分のパラメータがSC-88へ送信される。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

```
0(a1):REVERB MACRO      (00:Room1,01:Room2,02:Room3,
                         03:Hall1,04:Hall2,05:Plate,
                         06:Delay,07:Panning Delay(FB))
1(a1):REVERB CHARACTER                   ($00～$07)
2(a1):REVERB PRE-LPF                     ($00～$07)
3(a1):REVERB LEVEL                       ($00～$7F)
4(a1):REVERB TIME                        ($00～$7F)
5(a1):REVERB DELAY FEEDBACK              ($00～$7F)
6(a1):REVERB SEND LEVEL TO CHORUS($00～$7F/SC-88ではダミー)
7(a1):REVERB PRE-DELAY TIME              ($00～$7F)
(SC-88VLマニュアルp77参照)
```

ファンクション$48 ZM_SC88_CHORUS

機能 SC-88のコーラスパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:コーラスパラメータデータサイズ(1-9)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:コーラスパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にコーラスパラメータを記述しておく。d2.lは9以内で指定した個数分のパラメータがSC-88へ送信される。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
0(a1):CHORUS MACRO      (00:Chorus1,01:Chorus2,02:Chorus3,
                         03:Chorus4,04:Feedback Chorus,
                         05:Flanger,06:Short Delay,
                         07:Short Delay(FB))
1(a1):CHORUS PRE-LPF                     ($00～$07)
2(a1):CHORUS LEVEL                       ($00～$7F)
3(a1):CHORUS FEEDBACK                    ($00～$7F)
4(a1):CHORUS DELAY                       ($00～$7F)
5(a1):CHORUS RATE                        ($00～$7F)
6(a1):CHORUS DEPTH                       ($00～$7F)
7(a1):CHORUS SEND LEVEL TO REVERB        ($00～$7F)
8(a1):CHORUS SEND LEVEL TO DELAY         ($00～$7F)
(SC-88VLマニュアルp78参照)
```

ファンクション$49 ZM_SC88_DELAY

機能 SC-88のディレイパラメータの設定

引数 d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:ディレイパラメータデータサイズ(1-11)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:ディレイパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にリバーブパラメータを記述しておく。d2.lは11以内で、指定した個数分のパラメータがSC-88へ送信される。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

```
0(a1):DELAY MACRO       (00:Delay1,01:Delay2,02:Delay3,
                         03:Delay4,04:Pan Delay1,
                         05:Pan Delay2,06:Pan Delay3,
                         07:Pan Delay4,08:Delay to Reverb,
                         09:Pan Repeat)
1(a1):DELAY PRE-LPF                      ($00～$07)
2(a1):DELAY TIME CENTER             ($01～$73:0.1ms～sec)
3(a1):DELAY TIME RATIO LEFT              ($01～$78:4～500%)
4(a1):DELAY TIME RATIO RIGHT             ($01～$78:4～500%)
5(a1):DELAY LEVEL CENTER                 ($00～$7F)
6(a1):DELAY LEVEL LEFT                   ($00～$7F)
7(a1):DELAY LEVEL RIGHT                  ($00～$7F)
8(a1):DELAY LEVEL                        ($00～$7F)
9(a1):DELAY FEEDBACK                     ($00～$7F:-64～+63)
10(a1):DELAY SENDLEVEL TO REVERB ($00～$7F)
```

(SC-88VLマニュアルp78参照)

ファンクション$4a　ZM_SC88_EQUALIZER

機能　SC-88のイコライザパラメータの設定

引数　d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:イコライザパラメータデータサイズ(1-4)
d3.w:DEVICE ID(0-127,-1:以前の値を使用)
a1.l:イコライザパラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
d0.l=0以外:エラーコード

備考　a1.lからの領域にコーラスパラメータを記述しておく。d2.lは4以内で指定した個数分のパラメータがSC-88へ送信される。
d3.wを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
0(a1):EQ LOW FREQ              (0,1:200Hz,400Hz)
1(a1):EQ LOW GAIN              ($34～$4c:-12～+12dB)
2(a1):EQ HIGH FREQ             (0,1:3kHz,6kHz)
3(a1):EQ HIGH GAIN             ($34～$4c:-12～+12dB)
```

(SC-88VLマニュアルp78参照)

ファンクション$4b　ZM_SC88_PART_SETUP (ZM_SC88_PART_PARAMETER)

機能　SC-88のパートパラメータの設定

引数　d1.w:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
d2.l:パートパラメータデータサイズ(1-127)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:PART NUMBER(1-16)
a1.l:パートパラメータ格納バッファアドレス

戻り値　d0.l=0:正常終了
d0.l=0以外:エラーコード

備考　a1.lからの領域にパートパラメータを記述しておく。d2.lは127以内で指定した個数分のパラメータがSC-88へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
0(a1):Rx CHANNEL               (1～16,17=OFF)
1(a1):Rx PITCH BEND            (0～1:OFF/ON)
2(a1):Rx CH PRESSURE(CAf)      (0～1:OFF/ON)
3(a1):Rx PROGRAM CHANGE        (0～1:OFF/ON)
4(a1):Rx CONTROL CHANGE        (0～1:OFF/ON)
5(a1):Rx POLY PRESSURE(PAf)    (0～1:OFF/ON)
6(a1):Rx NOTE MESSAGE          (0～1:OFF/ON)
7(a1):Rx RPN                   (0～1:OFF/ON)
8(a1):Rx NRPN                  (0～1:OFF/ON)
9(a1):Rx MODURATION            (0～1:OFF/ON)
10(a1):Rx VOLUME               (0～1:OFF/ON)
11(a1):Rx PANPOT               (0～1:OFF/ON)
12(a1):Rx EXPRESSION           (0～1:OFF/ON)
13(a1):Rx HOLD1                (0～1:OFF/ON)
14(a1):Rx PORTAMENTO           (0～1:OFF/ON)
15(a1):Rx SOSTENUTO            (0～1:OFF/ON)
16(a1):Rx SOFT                 (0～1:OFF/ON)
17(a1):MONO/POLY MODE          (0～1:MONO/POLY)
18(a1):ASSIGN MODE  (0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
19(a1):USE FOR RHYTHM PART     (0:OFF,1:MAP1,2:MAP2)
20(a1):PITCH KEY SHIFT     ($28～$58:-24～+24[semitone])
21(a1):PITCH OFFSET FINE(upper) ($08～$F8:-12.0～+12.0[Hz])
22(a1):PITCH OFFSET FINE(lower)         //
23(a1):PART LEVEL              (0～127)
24(a1):VELOCITY SENSE DEPTH    (0～127)
25(a1):VELOCITY SENSE OFFSET   (0～127)
26(a1):PART PANPOT      (0:RANDOM,01(LEFT)～64～127(RIGHT))
27(a1):KEY RANGE LOW           (0～127:C-1～G9)
28(a1):KEY RANGE HIGH          (0～127:C-1～G9)
29(a1):CC1 CONTROL NUMBER      (0～127)
30(a1):CC2 CONTROL NUMBER      (0～127)
31(a1):CHORUS SEND LEVEL       (0～127)
32(a1):REVERB SEND LEVEL       (0～127)
33(a1):Rx BANK SELECT          (0:OFF,1:ON)
34(a1):Rx BANK SELECT LSB      (0:OFF,1:ON)

35(a1):PITCH FINE TUNE(Upper)
                    ($00 00～$40 00～$7f 7f:-100～0～+100)
36(a1):PITCH FINE TUNE(Lower)

37(a1):DELAY SEND LEVEL        (0～127)

38(a1):Vibrato rate            ($00～$7f:－64～＋63)
39(a1):Vibrato depth           ($00～$7f:－64～＋63)
40(a1):TVF cutoff freq.        ($00～$7f:－64～＋63)
41(a1):TVF resonance           ($00～$7f:－64～＋63)
42(a1):TVF&TVA Env.attack      ($00～$7f:－64～＋63)
43(a1):TVF&TVA Env.decay       ($00～$7f:－64～＋63)
44(a1):TVF&TVA Env.release     ($00～$7f:－64～＋63)
45(a1):Vibrato delay           ($00～$7f:－64～＋63)

46(a1):SCALE TUNING C        ($00～$7F:－64～＋63[cent])
47(a1):SCALE TUNING C#       ($00～$7F:－64～＋63[cent])
48(a1):SCALE TUNING D        ($00～$7F:－64～＋63[cent])
49(a1):SCALE TUNING D#       ($00～$7F:－64～＋63[cent])
50(a1):SCALE TUNING E        ($00～$7F:－64～＋63[cent])
51(a1):SCALE TUNING F        ($00～$7F:－64～＋63[cent])
52(a1):SCALE TUNING F#       ($00～$7F:－64～＋63[cent])
53(a1):SCALE TUNING G        ($00～$7F:－64～＋63[cent])
54(a1):SCALE TUNING G#       ($00～$7F:－64～＋63[cent])
55(a1):SCALE TUNING A        ($00～$7F:－64～＋63[cent])
56(a1):SCALE TUNING A#       ($00～$7F:－64～＋63[cent])
57(a1):SCALE TUNING B        ($00～$7F:－64～＋63[cent])

58(a1):MOD PITCH CONTROL    ($28～$58:－24～＋24[semitone])
59(a1):MOD TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
60(a1):MOD AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
61(a1):MOD LFO1 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
62(a1):MOD LFO1 PITCH DEPTH        ($00～$7F:0～600[cent])
63(a1):MOD LFO1 TVF DEPTH         ($00～$7F:0～2400[cent])
64(a1):MOD LFO1 TVA DEPTH          ($00～$7F:0～100.0[%])
65(a1):MOD LFO2 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
66(a1):MOD LFO2 PITCH DEPTH        ($00～$7F:0～600[cent])
67(a1):MOD LFO2 TVF DEPTH         ($00～$7F:0～2400[cent])
68(a1):MOD LFO2 TVA DEPTH          ($00～$7F:0～100.0[%])

69(a1):BEND PITCH CONTROL    ($40～$58:－24～＋24[semitone])
70(a1):BEND TVF CUTOFF CONTROL
                            ($00～$7F:－9600～＋9600[CENT])
71(a1):BEND AMPLITUDE CONTROL($00～$7F:－100.0～＋100.0[%])
72(a1):BEND LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
73(a1):BEND LFO1 PITCH DEPTH       ($00～$7F:0～600[cent])
74(a1):BEND LFO1 TVF DEPTH        ($00～$7F:0～2400[cent])
75(a1):BEND LFO1 TVA DEPTH         ($00～$7F:0～100.0[%])
76(a1):BEND LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
77(a1):BEND LFO2 PITCH DEPTH       ($00～$7F:0～600[cent])
78(a1):BEND LFO2 TVF DEPTH        ($00～$7F:0～2400[cent])
79(a1):BEND LFO2 TVA DEPTH         ($00～$7F:0～100.0[%])

80(a1):CAf PITCH CONTROL     ($28～$58:－24～＋24[semitone])
81(a1):CAf TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
82(a1):CAf AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
83(a1):CAf LFO1 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
84(a1):CAf LFO1 PITCH DEPTH        ($00～$7F:0～600[cent])
85(a1):CAf LFO1 TVF DEPTH         ($00～$7F:0～2400[cent])
86(a1):CAf LFO1 TVA DEPTH          ($00～$7F:0～100.0[%])
87(a1):CAf LFO2 RATE CONTROL  ($00～$7F:－10.0～＋10.0[Hz])
88(a1):CAf LFO2 PITCH DEPTH        ($00～$7F:0～600[cent])
89(a1):CAf LFO2 TVF DEPTH         ($00～$7F:0～2400[cent])
90(a1):CAf LFO2 TVA DEPTH          ($00～$7F:0～100.0[%])

91(a1):PAf PITCH CONTROL     ($28～$58:－24～＋24[semitone])
```

92(a1):PAf TVF CUTOFF CONTROL ($00～$7F:－9600～＋9600[CENT])
93(a1):PAf AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
94(a1):PAf LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
95(a1):PAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
96(a1):PAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
97(a1):PAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
98(a1):PAf LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
99(a1):PAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
100(a1):PAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
101(a1):PAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

102(a1):CC1 PITCH CONTROL ($28～$58:－24～＋24[semitone])
103(a1):CC1 TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
104(a1):CC1 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
105(a1):CC1 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
106(a1):CC1 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
107(a1):CC1 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
108(a1):CC1 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
109(a1):CC1 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
110(a1):CC1 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
111(a1):CC1 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
112(a1):CC1 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

113(a1):CC2 PITCH CONTROL ($28～$58:－24～＋24[semitone])
114(a1):CC2 TVF CUTOFF CONTROL($00～$7F:－9600～＋9600[CENT])
115(a1):CC2 AMPLITUDE CONTROL ($00～$7F:－100.0～＋100.0[%])
116(a1):CC2 LFO1 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
117(a1):CC2 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
118(a1):CC2 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
119(a1):CC2 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
120(a1):CC2 LFO2 RATE CONTROL ($00～$7F:－10.0～＋10.0[Hz])
121(a1):CC2 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
122(a1):CC2 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
123(a1):CC2 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

124(a1):TONE MAP NUMBER (0:SELECTED,1:SC-55,2:SC-88)
125(a1):TONE MAPO NUMBER (1:SC-55,2:SC-88)
126(a1):EQ ON/OFF (0:OFF,1:ON)
(SC-88VLマニュアルp79～p81参照)

ファンクション$4c ZM_SC88_DRUM_SETUP (ZM_SC88_DRUM_PARAMETER)

機能 SC-88のドラムセットアップ設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ドラムセットアップパラメータ・データサイズ(1-9)
d3.lw:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d3.hwlb:NOTE NUMBER(0-127)
d3.hwhb:MAP NUMBER(0-1)
a1.l:ドラムセットアップパラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にドラムセットアップパラメータを記述しておく。
d2.lは9以内で指定した個数分のパラメータがSC-88へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):PLAY KEY NUMBER (0～127)
1(a1):LEVEL (0～127)
2(a1):ASSIGN GROUP NUMBER (0:NON,1～127)
3(a1):PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
4(a1):REVERB DEPTH (0～127:0.0～1.0)
5(a1):CHORUS DEPTH (0～127:0.0～1.0)
6(a1):Rx NOTE OFF (0～1:OFF/ON)
7(a1):Rx NOTE ON (0～1:OFF/ON)
8(a1):DELAY SEND LEVEL (0-127:0.0～1.0)
(SC-88VLマニュアルp82参照)

ファンクション$4d ZM_SC88_DRUM_NAME

機能 SC-88のドラムセット名設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ドラムセットアップパラメータデータサイズ(1-8)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hw:MAP NUMBER(0,1,$80,$81)
a1.l:ドラムセット名文字列格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 文字列は12文字以内でa1.lからの領域に文字列を記述しておく。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
MAP NUMBER=$80がユーザードラムセット65,MAP NUMBER=$81がユーザードラムセット66に対応する。
(SC-88VLマニュアルp82参照)

ファンクション$4e ZM_SC88_USER_INST

機能 SC-88のユーザー音色の設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ユーザー音色パラメータデータサイズ(1-11)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hwlb:PROGRAM NUMBER(0-127)
d3.hwhb:BANK NUMBER(0-1)
a1.l:ユーザー音色パラメータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にユーザー音色パラメータを記述しておく。
d2.lは11以内で指定した個数分のパラメータがSC-88へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):SOURCE TONE#(MAP) (1,2)
1(a1):BANK NUMBER MSB (0～127)
2(a1):PROGRAM NUMBER (0～127)
3(a1):Vibrato rate ($00～$7f:－64～＋63)
4(a1):Vibrato depth ($00～$7f:－64～＋63)
5(a1):TVF cutoff freq. ($00～$7f:－64～＋63)
6(a1):TVF resonance ($00～$7f:－64～＋63)
7(a1):TVF&TVA Env.attack ($00～$7f:－64～＋63)
8(a1):TVF&TVA Env.decay ($00～$7f:－64～＋63)
9(a1):TVF&TVA Env.release ($00～$7f:－64～＋63)
10(a1):Vibrato delay ($00～$7f:－64～＋63)
(SC-88VLマニュアルp82参照)

ファンクション$4f ZM_SC88_USER_DRUM

機能 SC-88のユーザードラムセットの設定

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
d2.l:ユーザードラムセットパラメータデータサイズ(1-12)
d3.lw:DEVICE ID(0-127,-1:以前の値を使用)
d3.hwlb:NOTE NUMBER(0-127)
d3.hwhb:SET NUMBER(0-1)
a1.l:ユーザードラムセットパラメータ格納バッファ

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lからの領域にユーザードラムセットパラメータを記述しておく。
d2.lは12以内で指定した個数分のパラメータがSC-88へ送信される。
d3.lwを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

0(a1):PLAY KEY NUMBER (0～127)
1(a1):LEVEL (0～127)
2(a1):ASSIGN GROUP NUMBER (0:NON,1～127)
3(a1):PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
4(a1):REVERB DEPTH (0～127:0.0～1.0)
5(a1):CHORUS DEPTH (0～127:0.0～1.0)

6(a1):Rx NOTE OFF (0～1:OFF/ON)
7(a1):Rx NOTE ON (0～1:OFF/ON)
8(a1):DELAY SEND LEVEL (0-127:0.0～1.0)
9(a1):SOURCE DRUM SET#(MAP) (1-2)
10(a1):PROGRAM NUMBER (0-127)
11(a1):SOURCE NOTE NUMBER (0-127)
(SC-88VLマニュアルp82参照)

ファンクション$50 ZM_PCM_READ

機能 (AD)PCMファイルの登録

引数 d1.lw:登録番号(トーン番号の場合:0～32767)
(音色番号の場合:$8000+(0～32767)
d1.hw:PCM加工コマンド(PPC)でPCMデータ加工するか
(-1:YES,0:NO)
d2.hwhb:登録次元(-1:AD PCM,1=16BitPCM,0=V2互換)
d2.hwlb:オリジナルキー:0-127
a1.l:登録データ情報およびPPC列格納バッファ(備考参照)
0とするとd1.lwで表される登録済みのトーンや音色を削除

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
a0.l:a1.l処理後のアドレス
(ただし正常終了(d0.l=0)のときのみ有効)

備考 a1.lから、最大32文字分の音色名文字列を格納する。
終端コードは0。
次にディスクから(AD)PCMファイルを読みこむ場合はそのファイル名を格納する。終端コードは0。...(1)
ディスクからデータを読み込まずに、すでにZ-MUSIC側に登録されている(AD)PCMデータをコピーして加工するには音色名の後ろに$0000を格納しさらにそのあとに参照番号を指定する。...(2)
PPCで(AD)PCMを加工する場合には、PPC列をこの後ろに格納しておく。
また、PPCで(AD)PCMを加工する場合は必ずd1.hwを-1とすること。
d1.hwが0では、たとえPPCがあっても無視される。
PPCについてはMEASURE12を参照。

(1)の場合
[TONE NAME.. 最大32文字],0,[FILENAME文字列],0,PPCコマンド列...
(2)の場合
[TONE NAME...最大32文字],0,$00,$00,参照先番号(2bytes),PPCコマンド列...
参照先番号は、
ノート番号の場合:0～32767
音色番号の場合:$8000+(0～32767)
で表される。

ファンクション$51 ZM_REGISTER_ZPD

機能 ZPDの登録

引数 a1.l:登録するZPDのファイルネーム格納アドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

ファンクション$52 ZM_SET_ZPD_TABLE

機能 ZPDテーブルの登録

引数 d1.l:ZPDバージョンタイプ(0:V1.0～V2.0,1:V3,2:PDX)
a1.l:登録するZPDテーブルアドレス
(ZPDヘッダ「$1a,'ZmaDPcM'」含まず)

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 ZPDのテーブルの構造についてはMEASURE12を参照

ファンクション$53 ZM_CONVERT_PCM

機能 (AD)PCMデータの変換を行う

引数 d1.w:変換モード(0:AD PCM to PCM,1:PCM TO AD PCM)
d2.l:変換元の(AD)PCMのデータサイズ
a1.l:変換元となる(AD)PCMデータの格納領域アドレス
a2.l:変換先となる(AD)PCMデータの格納領域アドレス

戻り値 なし

ファンクション$54 ZM_EXEC_SUBFILE

機能 サブファイルの読み込みと実行

引数 a1.l:サブファイルのファイル名格納アドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 サブファイルとは演奏の際に必要となる別ファイルにまとめた補助ファイルのことを指す。具体的には、演奏ファイル(ZMS,ZMD,OPM)、AD PCM定義リスト(CNF),ZPDファイル、MIDIデータファイル(MDD,MID)など。

ファンクション$55 ZM_TRANSMIT_MIDI_DUMP

機能 MIDIデータファイルの転送

引数 d1.w:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
a1.l:MIDIデータファイルのファイル名格納アドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 MIDIデータファイルには、ver.2.0以前で標準MIDIデータファイルとされてきたMDDファイルとver.3.0から採用されたスタンダードMIDIファイル(FORMAT0)の2タイプがあるが、この関数ではどちらも楽器へ転送することができる。

ファンクション$56 ZM_SET_WAVE_FORM1

機能 波形メモリの登録1

引数 d1.w:波形番号(8-32767)
a1.l:波形メモリデータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lで表される波形メモリデータは以下の構成を取る。
0(a1):(.l)波形のデータサイズ(必ず偶数)
4(a1):(.b)ループタイプ(0:oneshot,1:repeat,2:altanative)
5(a1):(.l)ループ開始オフセット
9(a1):(.l)ループ終了オフセット
13(a1):(.l)ループ回数(0:∞,1～)
17(a1):(.l)ダミー
21(a1):(.b)波形名文字列長
22(a1):(.?):文字列(終端コード0:なし)
偶数アドレスから実際の波形データ(signed short)
:
:
※本ファンクションで登録された波形はZ-MUSICに取り込まれるため、登録後波形データ本体のメモリを解放しても構わない。

ファンクション$57 ZM_SET_WAVE_FORM2

機能 波形メモリの登録2

引数 d1.w:波形番号(8-32767)
a1.l:波形メモリデータ格納バッファアドレス

戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード

備考 a1.lで表される波形メモリデータは以下の構成はファンクションZM_SET_WAVE_FORM1と同じ
※本ファンクションで登録された波形はZ-MUSICに取り込まれないため、登録後波形データ本体のメモリを解放してはならない。

ファンクション$58 ZM_OBTAIN_EVENTS

機能 各種イベントの取得

引数 d1.w:どのイベントに関する設定を行うかのビットマーカー
ビット=1でそのイベントの設定を更新する。
a1.l:各種イベント発生時のジャンプ先とジャンプ条件が記述された領域。構成は備考参照。
-1とするとint omtによらず設定は一切行わない。

戻り値 a0.l=現在の設定を格納した領域のアドレス。構成は備考参照。

備考

$00(a1):m_play実行時のジャンプ先アドレス	d1.wのd0ビット
$04(a1):m_stop実行時のジャンプ先アドレス	d1.wのd1ビット
$08(a1):m_cont実行時ジャンプ先アドレス	d1.wのd2ビット
$0c(a1):演奏終了時のジャンプ先アドレス	d1.wのd3ビット
$10(a1):指定ループ回数(0:OFF,1-256)	d1.wのd4ビット
$14(a1):指定回ループ時のジャンプ先アドレス	d1.wのd5ビット
$18(a1):指定クロック(0:OFF,1～)	d1.wのd6ビット
$1c(a1):演奏開始後指定クロック経過時のジャンプ先アドレス	d1.wのd7ビット

d1.wでビット=1としたパラメータのみをa1.lから記述しておく。
d1.wでビット=0としたパラメータに関しては記述しない。
たとえばd1.w=$81の場合、a1.lからは、
$00(a1)に「m_playが実行されたときのジャンプ先アドレス」
$04(a1)に「演奏開始後指定クロック経過したときのジャンプ先
アドレス」
を記述しておけばよい。
なお、各ジャンプ先のルーチンではレジスタを破壊してはならず
(保存する必要がある)、ルーチンの終端は「RTS」命令で終わる
必要がある。

ファンクション$59 ZM_LOOP_CONTROL

機能 ループ制御
引数 d1.l:コントロールモード
-1=ループ回数取得
0=通常ループ指定
1=ループ無視指定
戻り値 d0.l=-1以外:現在のループ回数(1〜)
d0.l=-1:エラー
備考 ループ無視指定にいてはMEASURE14参照。

ファンクション$5a ZM_MASK_TRACKS

機能 トラックマスク
引数 a1.l:0=全トラックマスクモード
:1=トラックソロモード
:0,1以外=マスクするトラックとマスクモードのペアを格納した領域のアドレス
d1.w:a1.l=0のとき 0=マスク解除,1=マスク反転,-1=マスク設定
:a1.l=1のとき ソロ演奏させるトラック番号(0-65534)
:a1.l=0,1以外のとき無効
戻り値 a0.l=設定できなかったトラック番号(.w)の並びを格納した領域のアドレス。
終端コード=-1(.w)
備考 a1.l=0,1のとき、a1.lからは
トラック番号(.w):0-65534,
マスクモード(.w):0=MASK OFF, 1=REVERSE, -1=MASK ON
の数値列を記載する。終端コードは(short)-1。

ファンクション$5b ZM_MASK_CHANNELS

機能 チャンネルマスク
引数 a1.l:0=全トラックマスクモード
:1=トラックソロモード
:0,1以外=マスクするトラックとマスクモードのペアを格納した領域のアドレス
d1.w:a1.l=0のとき 0=マスク解除,1=マスク反転,-1=マスク設定
:a1.l=1のとき ソロ演奏させるチャンネルID
:a1.l=0,1以外のとき無効
戻り値 a0.l=設定できなかったチャンネルID(.l)の並びを格納したアドレス,
終端コード=-1(.w)
備考 a1.lからは、
チャンネルID(.l),マスクモード(.w)
の数値列を記載する。なお、チャンネルIDは、
チャンネルID(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI
チャンネルID(下位16ビット):チャンネル番号0-15
マスクモードは(0:MASK OFF,1:REVERSE,-1:MASK ON)の値をとる。
終端コードは-1(.w)。
たとえば、MIDI1の15チャンネル (ID:$8000_000e) をマスク(ID:$ffff)、AD PCMの3チャンネル(ID:$0001_0002)をマスク反転(ID:$0001)する場合、a1.lからの内容は
{ $8000,$000e,$ffff,$0001,$0002,$0001,$ffff }
となる。

ファンクション$5c ZM_SET_CH_OUTPUT_LEVEL

機能 各チャンネルの出力レベルの設定
引数 a1.l:出力パラメータを記載した領域のアドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lからは以下の構成の出力パラメータを格納しておく。
0(a1):(.w)デバイス(0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI)
2(a1):(.w)チャンネル番号(0-15)
4(a1):(.b)パラメータ有効ビット
(d0:移動速度,d1:開始レベル,2:終了レベル)
(対応ビット=1で、そのパラメータが後続する)
5(a1):(.w)移動速度(フェーダー移動速度。0で動かさない)
7(a1):(.b)フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
8(a1):(.b)フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

ファンクション$5d ZM_SET_TR_OUTPUT_LEVEL

機能 各トラックの出力レベルの設定
引数 a1.l:出力パラメータを記載した領域アドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lからは以下の構成の出力パラメータを格納しておく。
0(a1):(.w)トラック番号(0-65534)
2(a1):(.b)パラメータ有効ビット
(d0:移動速度,d1:開始レベル,2:終了レベル)
(対応ビット=1で、そのパラメータが後続する)
3(a1):(.w)移動速度(フェーダー移動速度。0で動かさない)
5(a1):(.b)フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
6(a1):(.b)フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

ファンクション$5e ZM_MASTER_FADER

機能 マスターフェーダーの設定
引数 a1.l:出力パラメータを記載した領域のアドレス
戻り値 d0.l=0:正常終了
d0.l=0以外:エラーコード
備考 a1.lからは以下の構成の出力パラメータを格納しておく。
0(a1):(.w)デバイス(0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI)
2(a1):(.b)パラメータ有効ビット
(d0:移動速度,d1:開始レベル,2:終了レベル)
(対応ビット=1で、そのパラメータが後続する)
3(a1):(.w)移動速度(フェーダー移動速度。0で動かさない)
5(a1):(.b)フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
6(a1):(.b)フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

ファンクション$5f ZM_GET_FADER_STATUS

機能 フェーダーの状態取得
引数 なし
戻り値 d0.lの第0,1,6,7ビットに検査結果が反映される。
d0:実行中のチャンネルフェーダーすべてが目的に達している
(1:達成した 0:達成していない)
d1:実行中のマスターフェーダーすべてが目的に達している
(1:達成した 0:達成していない)
d6:チャンネルフェーダーの状態(1:実行中 0:実行されていない)
d7:マスターフェーダーの状態(1:実行中 0:実行されていない)

ファンクション$60 ZM_GET_PLAY_TIME

機能 演奏時間の取得
引数 なし
戻り値 d0.lに16進表現で00HHMMSSという書式で返ってくる。
なおHH:0-99,MM:0-59,SS:0-59の値を取る。

ファンクション$61 ZM_GET_1ST_COMMENT

機能 演奏中の曲のコメントを取り出す
引数 なし
戻り値 a0.l=文字列のポインタ,終端コードは0

ファンクション$62 ZM_GET_TIMER_MODE

機能 現在のテンポソースとなっているタイマの種類を返す

引数　なし
戻り値　d0.l=0:タイマ A,1:タイマ B,2:YM3802タイマ

ファンクション$63 ZM_GET_TRACK_TABLE
機能　演奏トラックテーブルのアドレスを得る
引数　なし
戻り値　a0.l=音楽演奏トラックテーブルのアドレス
d0.l=効果音演奏トラックテーブルのアドレス
備考　音楽演奏トラックテーブル,効果音演奏トラックテーブルには演奏に使用されるトラック番号(0-65534)(.w)が-1(.w)を終端コードとして羅列される。

ファンクション$64 ZM_GET_PLAY_WORK
機能　音楽演奏トラックワークのアドレスを得る
引数　d1.w:トラック番号(0-65534)
戻り値　a0.l=音楽演奏トラックワークのアドレス
d0.l=効果音演奏トラックワークのアドレス
備考　音楽演奏トラックワーク,効果音演奏トラックの内容と構成についてはMEASURE13を参照のこと。

ファンクション$65 ZM_GET_BUFFER_INFORMATION
機能　バッファアドレステーブルのアドレス
引数　なし
戻り値　a0.l=バッファアドレステーブルのアドレス
備考　バッファアドレステーブルの構成についてはMEASURE13を参照のこと

ファンクション$66 ZM_GET_ZMSC_STATUS
機能　Z-MUSICの状態ワークのアドレス
引数　なし
戻り値　a0.l=Z-MUSICの状態ワークのアドレス
備考　Z-MUSICの状態ワークのアドレスの構成についてはMEASURE13を参照のこと

ファンクション$67 ZM_CALC_TOTAL
機能　ZMDの演奏時間やトータルステップタイムなどの計算
引数　a1.l:ZMDの先頭アドレス(ヘッダはあってもなくてもよい)
戻り値　d0.l=0:正常終了
a0.l=結果格納アドレス
d0.l=0以外:発生したエラーの数
a0.l=エラーテーブルアドレス
備考　・戻り値が0以外の場合はa0.lにはエラーの内容を格納したエラーテーブルのアドレスが返される。
エラーテーブルの構成についてはMEASURE11を参照のこと。
・いずれの場合もa0.lに返された領域は使用後ファンクションZM_FREE_MEMなどで解放すること。
・この関数はコンパイラに含まれる外部組み込みのファンクションとであるため、使用前にはコンパイラの常駐を確認する必要がある。
・戻り値が0の場合はa0.lには計算結果データ列を格納した領域のポインタが返される。計算結果データ列の構成を以下に示す。

```
TOTAL STEPTIME(.l)              グローバルリザルト群
TOTAL CHECKSUM(.l)
予想演奏時間(.l)                $00HHMMSS
RESERVED(.l)
RESERVED(.l)
RESERVED(.l)
RESERVED(.l)
RESERVED(.l)
TRACK1の情報までのOFFSET(.l)
TRACK2の情報までのOFFSET(.l)
        :
TRACKnの情報までのOFFSET(.l)
-1.(.w)                         End Code
TRACK1の情報
TRACK2の情報
        :
        :
TRACKxの情報構成
```

各トラックの情報は以下のような構成を取る

```
clc_total0(.l)          総ステップタイム(ループ外)
clc_total1(.l)          総ステップタイム(ループ内)
clc_checksum(.l)        総チェックサム
clc_measure(.l)         総小節数
```

以降、clc_measureで表される小節数分、以下の構成の情報が繰り返される。

```
clc_ms_total(.l)        第1小節の総ステップタイム
clc_dummy(.l)           RESERVED
clc_ms_checksum(.l)     第1小節のチェックサム
```

・自作プログラムから上記のオフセットテーブルを利用して計算結果情報を参照する場合はソースリストに含まれる ZCLABEL.MAC をインクルードするとよい。

ファンクション$68 ZM_APPLICATION_RELEASER
機能　外部アプリケーションの常駐解除ルーチンの登録
引数　a1.l:外部アプリケーションの常駐解除ルーチンの先頭アドレス
0に設定すると登録解除の指定となる。
d1.l:登録時に返されたリザルトコード(a1.l=0のときのみ有効)
戻り値　a0.l=0:エラー
a0.l=0以外:正常終了
d0.l:リザルトコード(ただしa0.l≠0のときのみ有効)
備考　・Z-MUSICが常駐解除するときに同時に解除できるようにするための手続き。Z-MUSICが常駐解除するときa1.lのルーチンにエントリする。
・d0.lには登録IDが返される。このIDは常駐解除ルーチンの登録をキャンセル(a1.l=0にて本ファンクションをコール)する場合に用いる。
・a1.lから始まる各解放ルーチンは必要処理を実行後、最後にZ-MUSIC側で解放してほしいメモリブロックがあるならばそのアドレスをa0.lに返すようにすること。なければa0.l=0を返すこと。
・a1.lからの解放ルーチンの実行後に、なにか表示したいメッセージがある場合はその文字列のアドレスをa1.lに返すようにすること。なければa1.l=0を返すこと。
・a1.lからの解放ルーチンでエラーが発生した場合は戻り値としてd0.l(0:no error(エラーなし),1:occupied(占有されている),-1:error(エラー発生))を返すことができる。
・エラーが発生した場合もその旨を告げるメッセージ文字列をa1.lに与えることができる。

ファンクション$69 ZM_RELEASE_DRIVER
機能　Z-MUSICの解除
引数　なし
戻り値　d0.l=0:正常終了
d0.l=0以外:エラーコード(常駐解除失敗)
a0.l=Z-MUSICの常駐アドレス
備考　戻り値a0.lで示される領域をシステムのメモリ解放コールMFREE(DOSCALL $FF49)を呼び出して解放した時点でZ-MUSICの常駐解除が完全に行われたことになる。

ファンクション$6a ZM_OCCUPY_ZMUSIC
機能　Z-MUSICの占有
引数　d1.l:1=占有する,0=占有を解除する,-1=占有状態の検査
戻り値　d0.l=以前の占有状態(0=占有されていない,1=占有されている)
(d1.l=0,1のときのみ有効)
d0.l=現在の占有状態(0=占有されていない,1=占有されている)
(d1.l=-1のときのみ有効)

ファンクション$6b ZM_HOOK_FNC_SERVICE
機能　Z-MUSICのファンクションコールのフック
引数　d1.l:ファンクション番号(0-127)
a1.l:ユーザーファンクションのエントリアドレス
-1とすると設定は行わずd1.lで指定したファンクション番号のファンクションコールのエントリ・アドレスのみを返す。
戻り値　a0.l=フック前のファンクションエントリアドレス

0ならば空いていた。
-1はエラー

ファンクション$6c ZM_HOOK_ZMD_SERVICE

機能 ZMDオペレーションのフック

引数 d1.lw:ZMDコード(0-255)
d1.hw:(0:FM,1:AD PCM,2:MIDI,
-1:共通コマンド,-2:効果音用共通コマンド,
-3:制御コマンド)
a1.l:ユーザーZMDオペレーションのエントリアドレス
-1とすると設定は行わずd1.lで指定したZMDコードの
オペレーションのエントリアドレスのみを返す。

戻り値 a0.l=フック前のZMDオペレーションエントリアドレス
0ならば空いていた。
-1はエラー

ファンクション$6d ZM_OCCUPY_COMPILER

機能 コンパイラの占有

引数 d1.l:1=占有,0=占有解除,-1=占有状態検査

戻り値 d0.l=以前の占有状態(0=占有されていない,1=占有されている)
(d1.l=0,1のときのみ有効)
d0.l=現在の占有状態(0=占有されていない,1=占有されている)
(d1.l=-1のときのみ有効)

ファンクション$6e ZM_STORE_ERROR

機能 エラーコードのストア

引数 d1.hw:エラーの発生したファンクション番号(0-32767)
d1.lw:エラーコード(0-65535)
d1.l=-1とした場合は問い合わせのみでストアは行わない
d2.l:エラーの合計個数値にいくつ加算するか

戻り値 d0.l=現在ストックしているエラーの個数
a0.l=エラーストックバッファのアドレス

備考 ・本関数はユーザーアプリケーション側で発生したエラーをZ-MUSIC側に知らせたい場合に使用する。ただしその場合、エラーコードフォーマットはMEASURE11で示してある形式に従わなければならない。
・エラーコードに関してはMEASURE11を参照。

ファンクション$6f ZM_PRINT_ERROR

機能 エラーメッセージの表示

引数 d1.hw:メッセージの出力先(0:画面,1:バッファ)
d1.lw:メッセージの言語(0:英語,1:日本語)
d2.l:エラーの数
a1.l:ZMDのファイル名(ない場合は0)
a2.l:ソース(ZMS)のアドレス(ない場合は0)
a3.l:ソース(ZMS)ファイル名(ない場合は0)
a5.l:エラーストックバッファの先頭アドレス

戻り値 d0.l=出力バッファのサイズ。ただし終端の0は含まない。
a0.l=出力バッファの先頭アドレス
(d1.hw=1のときのみ有効,バッファの終端には0)

備考 エラーテーブルからエラーメッセージを得る場合などに使用する。引数のa1.l～a3.lは通常の外部アプリケーションならば0でよい。これらを正しく設定すればタグジャンプに対応したエラーメッセージ出力を得ることができる。

ファンクション$70 ZM_GET_MEM

機能 メモリの確保

引数 d2.l:確保サイズ
d3.l:確保するメモリの用途ID

戻り値 d0.l≦0:確保に失敗した
a0.l=確保したメモリブロックのアドレス
(d0.l≧0のときのみ有効)

備考 Z-MUSICに無関係のメモリブロックを確保する場合ならば当ファンクションを用いる必要はない。
メモリの用途IDは以下のようなものがある。
自作プログラムから下記のメモリ用途IDを利用する場合はソースリストに含まれる ZMID.MAC をインクルードするとよい。

Z-MUSICメモリID	ID_ZMSC
ZPD	ID_ZPD
ZPDテーブル	ID_ZPD_TBL
ZMD	ID_ZMD
波形メモリ	ID_WAVE
波形メモリテーブル	ID_WAVE_TBL
AD PCM単音登録用(NOT ZPD)	ID_ADPCM
トータルステップタイム計算用ワークバッファ	ID_CALC
テンポラリ領域	ID_TEMP
外部アプリケーション解除登録ワーク	ID_APPLI
エラーストックバッファ	ID_ERROR
スタートアップファイル	ID_STARTUP
FM音源音色バッファ	ID_FMSND
システム関連ワーク	ID_SYSTEM
スタンダードMIDIファイル	ID_SMF
すべてを対象とする(FREE_MEM2時)	ID_ALL

ファンクション$71 ZM_ENLARGE_MEM

機能 メモリブロックのサイズの変更

引数 d2.l:変更後のサイズ
a1.l:ファンクションZM_GET_MEMで確保したメモリのアドレス

戻り値 d0.l≦0:確保に失敗した
a0.l=確保しなおしたメモリブロックのアドレス
(d0.l≧0のときのみ有効)

備考 ・メモリブロックのサイズを大きくした場合などはメモリブロックの再配置が自動で行われる。そのためa1.lで与えられたメモリブロックが別のアドレスに移動する場合がある。よってこのファンクションを実行後はa1.lのアドレス値を戻り値のa0.lのアドレス値に置き換えるべきである。
・変更に失敗した場合は、a1.lのメモリブロックの内容は保証される。

ファンクション$72 ZM_FREE_MEM

機能 メモリブロックの解放

引数 a1.l:ファンクションZM_GET_MEMで確保したメモリのアドレス

戻り値 d0.l=0:解放成功
d0.l=0以外:エラーコード

ファンクション$73 ZM_FREE_MEM2

機能 特定の用途IDを持ったメモリブロックの解放

引数 d3.l:メモリの用途ID(0は全IDのメモリブロック削除指定となる)

戻り値 d0.l=0:解放成功
d0.l=0以外:エラーコード

備考 メモリの用途IDはファンクションZM_GET_MEMの項を参照のこと。

ファンクション$74 ZM_EXCHANGE_MEMID

機能 メモリブロックの用途IDの変更

引数 d1.l:変更モード
(0:チェックなしで変更,1:チェックしてから更新
-1:d2.lの用途IDを持つメモリブロックをすべて
d3.lの用途IDへ変更する)
d2.l:更新元メモリの用途ID(d1.l=0の場合は意味なし)
d3.l:希望メモリの用途ID
a1.l:ファンクションZM_GET_MEMで確保したメモリのアドレス

戻り値 d0.l=0:変更成功
d0.l=0以外:エラーコード

備考 メモリの用途IDはファンクションZM_GET_MEMの項を参照のこと。

ファンクション$78 ZM_INIT_ALL

機能 Z-MUSICの全初期化を行う

引数 なし

戻り値 d0.l=Z-MUSICのバージョンID

備考 通常のアプリケーションは本関数を利用する必要はない
バージョンIDはたとえばバージョン3.21ならば $00003241が返る。
4の部分は固定値。

ファンクション$79 ZM_INT_START

機能 割り込み開始制御

引数 d1.w:制御対象タイマタイプ
(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)

戻り値 なし

備考 通常のアプリケーションは本関数を利用する必要はない

ファンクション$7a　ZM_INT_STOP

機能　割り込み停止制御

引数　d1.w:制御対象タイマタイプ
　(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)

戻り値　なし

備考　通常のアプリケーションは本関数を利用する必要はない

ファンクション$7b　ZM_SET_INT_SERVICE

機能　ユーザー割り込みサービスの設定

引数
・ユーザー割り込み登録
　d1.hw:全音符のクロック数(1-65535)
　d1.lw:テンポ(1-32767)
　a1.l:ユーザー割り込みルーチンのエントリアドレス
・登録したユーザー割り込みの割り込み速度(テンポ)を変更する
　d1.hw:全音符のクロック数(1-65535)
　d1.lw:テンポ(1-32767)
　a1.l:-1
・ユーザー割り込み状態検査
　d1.l:-1
　a1.l:=0
・ユーザー割り込み解除
　d1.l:0
　a1.l:ユーザー割り込みルーチンのエントリアドレス

戻り値
・ユーザー割り込み登録
　d0.l=0:登録正常終了
　d0.l=0以外:登録失敗
・登録したユーザー割り込みの割り込み速度(テンポ)を変更する
　d0.l=0:変更正常終了
　d0.l=0以外:変更失敗
・ユーザー割り込み状態検査
　d0.l=a0.l=0:空き
　d0.l=-1,a0.l=先約ユーザー割り込みエントリアドレス
・ユーザー割り込み解除
　d0.l=0:解除正常終了
　d0.l=0以外:解除失敗

備考
・割り込みルーチンの最後はRTEではならない。必ず全レジスタを保存したうえでRTSで終了帰還するような作りにすること。
・引数のa1.l=-1とした場合はすでに登録されている割り込みサービスの割込み周期をd1.lw,d1.hwのテンポ情報に変更する機能を果たす。

ファンクション$7c　ZM_CONTROL_TEMPO

機能　テンポ設定の主導権設定

引数　d1.l:-1=テンポ設定の主導権を外部アプリケーションに委託
　0=テンポ設定の主導権をZ-MUSICに戻す
　(このとき内部パラメータでテンポの初期化も実施する)

戻り値　d0.l=0:設定正常終了
　d0.l=0以外:設定失敗

ファンクション$7f　ZM_EXEC_ZMD

機能　ZMD列の実行

引数　d1.l=トラック番号(0-65534,65535:特に指定しない場合)
　d2.l=ZMD列の長さ
　a1.l=ZMD列格納アドレス
　(バッファ自体はd2.l+8の大きさを必要とする)

戻り値　なし

MEASURE 11

エラー

ここではエラー発生時のエラーメッセージ、エラーの原因と対処方法について解説します。

11．1．　Z-MUSIC ver.3.0のエラー

ZMC.XによるZMSコンパイル時のエラー、そしてZPCNV3.RのCNFファイル→ZPD変換時のエラーは以下に示すエラーメッセージによってその症状が報告される。

これらのエラーひとつひとつには、固有のエラーIDが割り当てられており、ZMSC3.Xのファンクションコール実行時に発生したエラーコードのエラー番号と一致する。

エラーID:0
(ENGLISH):OUT OF MEMORY
(日本語)：メモリが不足しています
原因と対処:メモリを増設するか、不要な常駐ソフトを常駐解除し、フリーエリアを広げる。

エラーID:1
(ENGLISH):UNIDENTIFIED FILE
(日本語)：ZMUSICシステムのファイルではありません
原因と対処:本当にそのファイルがZ-MUSIC ver.3.0で使用できるかを確認する。

エラーID:2
(ENGLISH):ILLEGAL VERSION NUMBER
(日本語)：バージョン番号が違います
原因と対処:Z-MUSIC ver3.0ではver.2.0のZMDなどは演奏できない。

エラーID:3
(ENGLISH):ILLEGAL FILENAME
(日本語)：ファイル名が異常です
原因と対処:ファイル名に規定外の文字が含まれていないか確認する。

エラーID:4
(ENGLISH):MISSING FILENAME
(日本語)：ファイル名がありません
原因と対処:ファイル名を正しく指定する。

エラーID:5
(ENGLISH):FILE NOT FOUND
(日本語)：ファイルが見つかりません

エラーID:6
(ENGLISH):ILLEGAL FILE SIZE
(日本語)：ファイルのサイズが異常です
原因と対処:Z-MUSIC ver.3.0ではメインメモリサイズを超えるようなファイルは読み込むことができない。

エラーID:7
(ENGLISH):READ ERROR
(日本語)：ディスクからの読み込みに失敗しました
原因と対処:ディスクやファイルが破壊されていないか確認する。

エラーID:8
(ENGLISH):TOO MANY INCLUDE FILES
(日本語)：インクルードファイルが多すぎます

エラーID:9
(ENGLISH):NO ZMD ERROR
(日本語)：汎用バッファ内にはZMUSICが扱えるZMDがありません

エラーID:10
(ENGLISH):UNIDENTIFIED MEMORY
(日本語)：ZMUSICシステムに無関係のメモリブロックです

エラーID:11
(ENGLISH):COMPILER NOT AVAILABLE
(日本語)：コンパイラが常駐していないのでコンパイルは出来ません
原因と対処:ZMC.Xを常駐させる。

エラーID:12
(ENGLISH):COMPILE ERROR
(日本語)：コンパイラでエラーが発生しました

エラーID:13
(ENGLISH):NO APPLICATION REGISTERED
(日本語)：アプリケーションは登録されていません

エラーID:14
(ENGLISH):ILLEGAL RESULT CODE
(日本語)：リザルトコードが異常です

エラーID:15
(ENGLISH):ILLEGAL ZMD CODE
(日本語)：規定外のZMDを参照しました

エラーID:16
(ENGLISH):ILLEGAL FUNCTION NUMBER
(日本語)：規定外のファンクションコールを参照しました

エラーID:17
(ENGLISH):NO PERFORMANCE DATA
(日本語)：演奏データが存在しません

エラーID:18
(ENGLISH):UNMANAGEABLE DATA SEQUENCE
(日本語)：未対応のデータ構造です

エラーID:19
(ENGLISH):THE DEVICE ALREADY OCCUPIED
(日本語)：すでにそのデバイスは占有されています

エラーID:20
(ENGLISH):RECUSIVE INCLUDE ERROR
(日本語)：同じファイルをインクルードしようとしました

エラーID:23
(ENGLISH):ILLEGAL CHANNEL
(日本語):規定外のチャンネル番号を指定しました

エラーID:27
(ENGLISH):ILLEGAL TRACK NUMBER
(日本語):規定外のトラック番号を指定しました
原因と対処:Z-MUSIC ver.3.0で取り扱うことができるトラックはトラック番号1〜65535。

エラーID:28
(ENGLISH):TRACK COMMAND ERROR
(日本語):演奏データ登録コマンドの使用法に誤りがあります

エラーID:29
(ENGLISH):TOO MANY TRACKS
(日本語):トラック数が多すぎます
原因と対処:Z-MUSIC ver.3.0で取り扱うことができるトラック数は65535本まで。

エラーID:30
(ENGLISH):ILLEGAL TRACK SIZE
(日本語):トラックサイズが異常です

エラーID:31
(ENGLISH):ILLEGAL TRACK VOLUME
(日本語):トラック音量の値が異常です
原因と対処:トラック音量の有効範囲は0-256。

エラーID:32
(ENGLISH):ILLEGAL TRACK FREQUENCY
(日本語):トラック処理頻度値が異常です
原因と対処:トラック処理頻度値の有効範囲は1-256。

エラーID:33
(ENGLISH):UNDEFINED TRACK REFERRED
(日本語):未定義のトラックを参照しました

エラーID:34
(ENGLISH):TRACK NUMBER REDESIGNATION
(日本語):同一のトラックが重複して指定されています

エラーID:38
(ENGLISH):PATTERN NAME REDEFINITION
(日本語):パターン名が重複しています

エラーID:39
(ENGLISH):PATTERN NAME TOO LONG
(日本語):パターン名が長すぎます

エラーID:40
(ENGLISH):PATTERN NAME CANNOT BE OMITTED
(日本語):パターン名は省略できません

エラーID:41
(ENGLISH):NULL PATTERN ERROR
(日本語):空のパターンは定義できません

エラーID:42
(ENGLISH):PATTERN NOT AVAILABLE
(日本語):パターンは使用できません

エラーID:43
(ENGLISH):UNDEFINED PATTERN
(日本語):未定義のパターンを指定しました

エラーID:44
(ENGLISH):ILLEGAL COMMAND IN PATTERN
(日本語):パターンに含められないコマンドを使用しました
原因と対処:繰り返しコマンドはパターン中に含められない。

エラーID:48
(ENGLISH):ILLEGAL BANK NUMBER
(日本語):規定外のバンク番号を指定しました
原因と対処:FM音源,(AD)PCM音源パートでのバンク番号の有効範囲は0-255。MIDI音源パートでは0-16383。

エラーID:49
(ENGLISH):ILLEGAL TIMBRE NUMBER
(日本語):規定外の音色番号を指定しました
原因と対処:FM音源,(AD)PCM音源パートでの音色番号の有効範囲は1-32768。MIDI音源パートでは1-128。

エラーID:50
(ENGLISH):ILLEGAL TONE NUMBER
(日本語):規定外のトーン番号を指定しました
原因と対処:トーン番号の有効範囲は0-127。

エラーID:51
(ENGLISH):NO TIMBRE PARAMETERS
(日本語):FM音源音色のパラメータはありません

エラーID:52
(ENGLISH):ILLEGAL TIMBRE PARAMETER
(日本語):規定外の音色パラメータが設定されています

エラーID:53
(ENGLISH):TIMBRE PARAMETER SHORTAGE
(日本語):音色パラメータが不足しています

エラーID:54
(ENGLISH):TIMBRE NAME TOO LONG
(日本語):音色名が長すぎます

エラーID:55
(ENGLISH):EMPTY TIMBRE NUMBER
(日本語):空の音色番号を参照しました

エラーID:59
(ENGLISH):ILLEGAL NOTE NUMBER
(日本語):規定外のノートナンバーを指定しました
原因と対処:ノートナンバーの有効範囲は0-127。

エラーID:60
(ENGLISH):ILLEGAL NOTE LENGTH
(日本語):音長値が規定範囲外です
原因と対処:音長値の有効範囲はマスタークロックの値に左右される。

エラーID:61
(ENGLISH):EMPTY NOTE NUMBER
(日本語):空のノート番号を参照しました

エラーID:62
(ENGLISH):TOO MANY NOTES
(日本語):音符が多すぎます
原因と対処:和音内に含められる音符数は16以内。

エラーID:66
(ENGLISH):PROCESSING SIZE TOO LARGE
(日本語):加工処理サイズが大きすぎます

エラーID:67
(ENGLISH):UNDEFINED PPC
(日本語):未定義のPCM加工コマンドです

エラーID:68
(ENGLISH):UNDEFINED LOOP TYPE
(日本語):未定義のADPCM/PCMループタイプが指定されています

エラーID:69
(ENGLISH):ILLEGAL FREQUENCY VALUE

(日本語):周波数の値が規定外です
原因と対処:周波数の値の有効範囲は1-65535[Hz]。

エラーID:70
(ENGLISH):CUT OFF LEVEL TOO BIG
(日本語):切り捨てレベルが大きすぎます
原因と対処:切り捨てレベルの有効範囲は0-32767。

エラーID:71
(ENGLISH):WAVE COMMAND ERROR
(日本語):波形メモリ登録コマンドエラーです

エラーID:72
(ENGLISH):ILLEGAL WAVE NUMBER
(日本語):波形番号が異常です
原因と対処:プリセット波形は波形番号0-7、ユーザー波形(波形メモリ)は波形番号0-255で指定される。

エラーID:73
(ENGLISH):ILLEGAL WAVE VALUE
(日本語):波形メモリ値が規定外です
原因と対処:波形メモリ構成値の有効範囲は-32768～+32767。

エラーID:74
(ENGLISH):ILLEGAL LOOP START POINT
(日本語):ループ開始ポイントの位置が異常です

エラーID:75
(ENGLISH):ILLEGAL LOOP END POINT
(日本語):ループ終了ポイントの位置が異常です

エラーID:79
(ENGLISH):DEVICE OFFLINE
(日本語):そのデバイスは未接続です

エラーID:80
(ENGLISH):ILLEGAL INTERFACE NUMBER
(日本語):インターフェース番号(名前)が規定外です

エラーID:81
(ENGLISH):ILLEGAL DEVICE ID
(日本語):デバイスIDの値が規定外です

エラーID:82
(ENGLISH):DEVICE ID REDESIGNATION
(日本語):同一デバイスが重複して指定されています

エラーID:86
(ENGLISH):ILLEGAL PART NUMBER
(日本語):パート番号が異常です

エラーID:87
(ENGLISH):ILLEGAL MODEL ID
(日本語):モデルIDの値が規定外です

エラーID:88
(ENGLISH):ILLEGAL MAKER ID
(日本語):メーカーIDの値が規定外です

エラーID:89
(ENGLISH):ILLEGAL MAP NUMBER
(日本語):マップ番号が規定外です

エラーID:90
(ENGLISH):ILLEGAL PARTIAL NUMBER
(日本語):パーシャル番号が規定外です

エラーID:91
(ENGLISH):ILLEGAL PATCH NUMBER
(日本語):パッチ番号が規定外です

エラーID:95
(ENGLISH):SYNTAX ERROR
(日本語):文法エラーです

エラーID:96
(ENGLISH):ASSIGN COMMAND ERROR
(日本語):チャンネル/トラック・割り当てコマンドの使用法に誤りがあります

エラーID:97
(ENGLISH):ILLEGAL OCTAVE
(日本語):規定外のオクターブを指定しました
原因と対処:オクターブの有効範囲は-1～9

エラーID:98
(ENGLISH):ILLEGAL PITCH VALUE
(日本語):ピッチの値が規定外です

エラーID:99
(ENGLISH):ILLEGAL VOLUME VALUE
(日本語):音量の値が規定外です
原因と対処:音量の有効範囲は0-16、絶対音量の有効範囲は0-127。

エラーID:100
(ENGLISH):ILLEGAL VELOCITY VALUE
(日本語):ベロシティの値が規定外です
原因と対処:ベロシティの有効範囲は0-127。

エラーID:101
(ENGLISH):ILLEGAL AFTERTOUCH VALUE
(日本語):アフタータッチの値が規定外です
原因と対処:アフタータッチの値の有効範囲は0-127。

エラーID:102
(ENGLISH):ILLEGAL PANPOT VALUE
(日本語):パンポットの値が規定外です
原因と対処:パンポットの値の有効範囲は0-127。

エラーID:103
(ENGLISH):ILLEGAL DAMPER VALUE
(日本語):ダンパーの値が規定外です
原因と対処:ダンパーの値の有効範囲は0-127。

エラーID:107
(ENGLISH):ILLEGAL COMMAND LINE
(日本語):コマンド書式に誤りがあります

エラーID:108
(ENGLISH):ILLEGAL COMMAND ORDER
(日本語):コマンドの使用順序が規定外です

エラーID:109
(ENGLISH):INAPPROPRIATE COMMAND
(日本語):不適切なコマンドが使用されています

エラーID:110
(ENGLISH):PARAMETER BREAK OFF
(日本語):パラメータが途中で終わっています

エラーID:111
(ENGLISH):PARAMETER SHORTAGE
(日本語):パラメータが少なすぎます

エラーID:112
(ENGLISH):MISSING OPERAND
(日本語):パラメータがあるべきところに有りません

エラーID:113
(ENGLISH):UNEXPECTED OPERAND
(日本語):予期しないパラメータが設定されました

エラーID:114
(ENGLISH):ILLEGAL PARAMETERS COMBINATION
(日本語)：パラメータの組み合わせが規定外です

エラーID:115
(ENGLISH):REASSIGNMENT ERROR
(日本語)：同一トラックを複数のチャンネルに割り当てようとしました
原因と対処:同一チャンネルを複数のトラックに割り当てることはできるが、同一トラックを複数のチャンネルに割り当てることはできない。

エラーID:116
(ENGLISH):ILLEGAL NESTING ERROR
(日本語)：括弧の対応に誤りがあります

エラーID:117
(ENGLISH):PARAMETER CANNOT BE OMITTED
(日本語)：このパラメータの省略はできません

エラーID:118
(ENGLISH):COMMAND LINE BREAK OFF
(日本語)：コマンドラインが途中で終了しています

エラーID:119
(ENGLISH):ILLEGAL OPERAND
(日本語)：規定外のオペランドです

エラーID:120
(ENGLISH):ILLEGAL PARAMETER FORMAT
(日本語)：パラメータのフォーマットが規定外です

エラーID:124
(ENGLISH):EMPTY BRACE ERROR
(日本語)：{}の中身が空です

エラーID:125
(ENGLISH):ILLEGAL COMMAND IN BRACE
(日本語)：{}の中には含められないコマンドを使用しました

エラーID:126
(ENGLISH):TOO MANY PARAMETERS
(日本語)：パラメータの数が多すぎます

エラーID:127
(ENGLISH):GROUP NOTES COMMAND ERROR
(日本語)：連符コマンドの使用法に誤りがあります

エラーID:131
(ENGLISH):KANJI BREAK OFF
(日本語)：全角文字が1バイトのみで終わっています

エラーID:132
(ENGLISH):ILLEGAL STRING LENGTH
(日本語)：文字列の長さが異常です

エラーID:133
(ENGLISH):STRING BREAK OFF
(日本語)：文字列の終端が見つかりません

エラーID:134
(ENGLISH):STRING TOO LONG
(日本語)：文字列が長すぎます

エラーID:138
(ENGLISH):ILLEGAL DATA SIZE
(日本語)：データサイズが異常です

エラーID:139
(ENGLISH):DELAY TOO LONG
(日本語)：ディレイが長すぎます

エラーID:140
(ENGLISH):ILLEGAL DELAY
(日本語)：ディレイが規定外です

エラーID:141
(ENGLISH):HOLD TIME TOO LONG
(日本語)：ホールドタイムが長すぎます

エラーID:142
(ENGLISH):BEND TIME TOO LONG
(日本語)：ピッチベンドタイムが長すぎます

エラーID:143
(ENGLISH):ILLEGAL BEND RANGE
(日本語)：ピッチベンド幅が有効範囲を超えています

エラーID:144
(ENGLISH):OFFSET TOO LONG
(日本語)：オフセットが長すぎます

エラーID:145
(ENGLISH):SIZE TOO LARGE
(日本語)：サイズが大きすぎます

エラーID:146
(ENGLISH):ILLEGAL PARAMETER VALUE
(日本語)：データが規定外です

エラーID:147
(ENGLISH):ILLEGAL CHARACTER
(日本語)：この文字は使用できません

エラーID:148
(ENGLISH):ILLEGAL SWITCH VALUE
(日本語)：規定外のスイッチを設定しようとしました

エラーID:149
(ENGLISH):ILLEGAL CATEGORY EVENT
(日本語)：規定外の種類のイベントを設定しようとしました

エラーID:150
(ENGLISH):UNKNOWN EVENT CLASS
(日本語)：未知のイベントデータを設定しました

エラーID:151
(ENGLISH):KEY TRANSPOSE OUT OF RANGE
(日本語)：キートランスポーズが有効範囲を超えて設定されました

エラーID:152
(ENGLISH):ILLEGAL EFFECT PARAMETER
(日本語)：エフェクターへのパラメータが規定外です

エラーID:153
(ENGLISH):ILLEGAL NOISE PARAMETER
(日本語)：ノイズパラメータが規定外です

エラーID:154
(ENGLISH):UNDEFINED ZMD CODE
(日本語)：未定義のZMDです

エラーID:155
(ENGLISH):ERROR IN DIVISION
(日本語)：音長計算の割り算で商が0になりました

エラーID:156
(ENGLISH):ILLEGAL TIME VALUE
(日本語)：時間の値が規定外です

エラーID:160
(ENGLISH):TEMPO COMMAND ERROR
(日本語):テンポコマンドの使用法に誤りがあります

エラーID:161
(ENGLISH):ILLEGAL TEMPO VALUE
(日本語):テンポの値が規定外です

エラーID:163
(ENGLISH):ILLEGAL REPEAT TIME
(日本語):繰り返し回数が規定外です

エラーID:164
(ENGLISH):DISORDERLY REPEAT STRUCTURE
(日本語):繰り返し構造が異常です

エラーID:167
(ENGLISH):TOO MANY SIGNS
(日本語):臨時記号が多すぎます
原因と対処:臨時記号は最大7つまで。

エラーID:168
(ENGLISH):ILLEGAL SIGN
(日本語):未知の臨時記号が設定されました

エラーID:169
(ENGLISH):UNKNOWN KEY DECLARED
(日本語):未知の調号が設定されました

エラーID:173
(ENGLISH):PORTAMENT TIME TOO LONG
(日本語):ポルタメントタイムが長すぎます
原因と対処:ポルタメントタイムの有効範囲は1～32767。

エラーID:174
(ENGLISH):PORTAMENT COMMAND ERROR
(日本語):ポルタメントコマンドの使用法に誤りがあります

エラーID:177
(ENGLISH):CHORD COMMAND ERROR
(日本語):和音コマンドの使用法に誤りがあります

エラーID:181
(ENGLISH):ILLEGAL REGISTER NUMBER
(日本語):レジスタ番号が規定外です

エラーID:182
(ENGLISH):UNKNOWN REGISTER NAME
(日本語):未知のレジスタ名を指定しました

エラーID:183
(ENGLISH):ILLEGAL CONTROL NUMBER
(日本語):規定外のコントロールを指定しました

エラーID:186
(ENGLISH):ILLEGAL MODE VALUE
(日本語):モード値が規定外です

エラーID:187
(ENGLISH):UNDEFINED MODE
(日本語):未定義のモードを設定しようとしました

エラーID:188
(ENGLISH):ILLEGAL TIE MODE
(日本語):規定外のタイモードを設定しようとしました

エラーID:189
(ENGLISH):ILLEGAL RESERVATION
(日本語):規定外の予約を行おうとしました

エラーID:193
(ENGLISH):ILLEGAL DEPTH VALUE
(日本語):振幅が規定外です

エラーID:194
(ENGLISH):SPEED TOO SLOW
(日本語):スピードが遅すぎます

エラーID:195
(ENGLISH):ILLEGAL SPEED VALUE
(日本語):スピードが規定外です

エラーID:196
(ENGLISH):ILLEGAL ARCC CONTROL
(日本語):規定外のARCCコントロールを設定しました
原因と対処:ARCCコントロール値の有効範囲は0-127。

エラーID:197
(ENGLISH):ILLEGAL RESET VALUE
(日本語):リセット値が規定外です
原因と対処:リセット値の有効範囲は0-127。

エラーID:198
(ENGLISH):ILLEGAL WAVE ORIGIN
(日本語):規定外の波形の基準点を設定しました
原因と対処:波形の基準点の有効範囲は0-127。

エラーID:199
(ENGLISH):ILLEGAL PHASE TYPE
(日本語):未定義の位相タイプを指定しました
原因と対処:位相タイプはNORMALとREVERSE(INVERSE)のいずれか

エラーID:202
(ENGLISH):ILLEGAL FADER LEVEL
(日本語):規定外のフェーダーレベルを設定しようとしました
原因と対処:フェーダーレベルの有効範囲は0-128。

エラーID:203
(ENGLISH):ILLEGAL FADER SPEED
(日本語):規定外のフェーダースピードを設定しようとしました
原因と対処:フェーダースピードの有効範囲は0-255。

エラーID:207
(ENGLISH):ILLEGAL MASTER CLOCK
(日本語):マスタークロックの値が規定外です
原因と対処:マスタークロックの値の有効範囲は1-32767。

エラーID:208
(ENGLISH):ILLEGAL METER
(日本語):規定外の拍子を設定しようとしました
原因と対処:拍子の有効範囲は1/1～255/255。

エラーID:211
(ENGLISH):MACRO NAME TOO LONG
(日本語):マクロ名が長すぎます
原因と対処:マクロ名は'{'を除く英数字、全角の240文字以内の文字列で与える。

エラーID:212
(ENGLISH):MACRO NAME CANNOT BE OMITTED
(日本語):マクロ名は省略できません

エラーID:216
(ENGLISH):ILLEGAL GATE RANGE
(日本語):規定外のゲートレンジを設定しようとしました
原因と対処:ゲートタイムレンジは8,16,32,64,128のいずれかでなければならない。

エラーID:217
(ENGLISH):ILLEGAL GATE TIME

(日本語):規定外のゲートタイムを設定しようとしました
原因と対処:ゲートタイムの有効範囲は0-32767。

以下はエラーではなくてウォーニング(警告)。
ウォーニングの場合はコンパイル結果であるZMDは出力されるものの、その内容は完全に保証されるものではない。

エラーID:220
(ENGLISH):SURPLUS IN DIVISION
(日本語):音長計算の割り算であまりが発生しました
原因と対処:そのトラックが他のトラックに対して演奏がずれてくる可能性がある。

エラーID:221
(ENGLISH):ZMD DIRECTLY EMBEDDED
(日本語):ZMDを直接埋め込みました

エラーID:222
(ENGLISH):ILLEGAL FREQUENCY NUMBER
(日本語):周波数番号が規定外です
原因と対処:MML @Fで与えられる周波数番号の有効範囲は0-6。

エラーID:223
(ENGLISH):SYNC AND JUMP CONTROL ARE USED AT THE SAME TIME
(日本語):同期制御とジャンプ制御を同時に行っています
原因と対処:各トラックの演奏がずれてくる可能性がある。

エラーファイルの活用

コンパイル時のエラーメッセージをうまく活用すると開発効率が上がります。たとえば、以下のようにして'MUSIC.ZMS'というZMSをコンパイルした場合、

A>ZMc MUSIC > ER

エラーが発生すると'ER'というファイルネームのファイルができます(これをエラーファイルと呼びます)。次に、ここで

A>ED ER

としてエディタにエラーファイルを読み込んでください。エラーファイルの読み込みを確認したら[ESC]を押してから[V]を押してください。するとMUSIC.ZMSファイルを読み込み、'MUSIC.ZMS'のエラーの発生した行へとカーソルがジャンプします(これはエディタのタグジャンプという機能です)。後はエディタで誤った箇所を訂正するだけです。エラーが複数あった場合は[SHIFT]+[F6]で、もう一度エラーファイルの画面にして次のエラーメッセージの行にカーソルを動かして再び[ESC]+[V]を実行してみましょう。すると'MUSIC.ZMS'ファイルのそのエラーのある行へジャンプしているはずです。
こうしてエラーを修正していきます。エラーが全くない場合はエラーファイルは「空(NULL)」ファイルになります。
なお、エラーファイルは、

MUSIC.ZMS　　18　　SYNTAX ERROR

のようにファイル名、行番号、エラーメッセージの順番になっています。

11．2．　ZMSC3.Xファンクションコールのエラー

ZMSC3.Xのファンクションコールの実行中、エラーが発生すると、戻り値はa0.l=0になり、d0.lにエラーコードを返す。このエラーコードは($8000+エラーの発生したファンクション番号(0-32767))*65536+エラーID(0-65535)で表される。エラーが、ZMSC3.X内の、ファンクションコール外の処理系で発生した場合はエラーコードのファンクション番号は32767となる。

例

copy filename.ZMS ZMS

実行後エラー発生の場合など

ファンクションZM_COMPILERではエラー発生時、エラーコードのほかに付随した情報を持ったエラーテーブルをa0.lに返す。ファンクションZM_COMPILER内で複数のエラーが発生した場合、以下の構成をとる合計16バイトのエラー情報がそのエラー発生個数分a0.lより格納される。

エラーコード(.l),付加情報1(.l),付加情報2(.l),付加情報3(.l)
合計16バイト

なお、後半3ロングワード分の付加情報はエラーIDごとに異なり、以下のような構成をとる。

●通常の場合

付加情報1=ファイル名格納アドレス(.l)
付加情報2=ソースファイルにおけるエラー発生行(.l),
付加情報3=桁位置(.l)

付加情報1は通常は0。インクルードしたファイルなどでエラーが発生した場合のみ有効なアドレスをとる。付加情報1=0は読み込んだ親ソースを表す。

●エラー「FILE_NOT_FOUND」「ILLEGAL_FILE_SIZE」「READ_ERROR」の場合

付加情報1=ファイル名格納アドレス(.l)
付加情報2=0(.l)
付加情報3=0(.l)

●エラー「UNDEFINED_ZMD_CODE」の場合

付加情報1=ファイル名格納アドレス(.l)
付加情報2=未定義のZMDが存在するオフセットアドレス(.l)
付加情報3=未定義ZMD(.l)

また、ユーザーアプリケーションなどで、このようなエラーテーブルを扱う場合、特に付加情報を持たない場合は付加情報1～3は0となる。
エラーテーブルは利用後、ファンクションコールZM_COMPILERを呼び出したアプリケーションが責任を持って解放しなければならない。そのときにはファンクションZM_FREE_MEMなどのZ-MUSICファンクションコールを使用しなければならない。その他の方法で解放した場合はその後のZ-MUSICの動作が保証できなくなる。

11．3．　ZMSC3．Xのエラー

(ENGLISH):An application which cannot cooperate with ZMUSIC has been included.
(日本語):協調不可能のアプリケーションが常駐しています.
原因と対処:ZMSC3.Xが占有したい割り込みベクタなどがすでに使用されている。

(ENGLISH):ZMUSIC PERFORMANCE MANAGER was already included.
(日本語):ZMUSIC演奏マネージャはすでに常駐しています.

(ENGLISH):An unidentified ADPCM driver has been included.
(日本語):認識不可能のADPCMドライバが常駐しています.
原因と対処:Z-MUSIC ver.3.0が対応するAD PCMドライバはMPCM.Xおよび、これの互換ドライバに限られる。

(ENGLISH):MPCM denies ZMUSIC occupation.
(日本語):MPCMがZMUSICによる占有を拒否しました.
原因と対処:MPCM.Xがほかのアプリケーションに占有されているためZMSC3.Xが利用できない。

(ENGLISH):Out of memory error.
(日本語):メモリが不足しています.
原因と対処:メモリを増設するか、不要な常駐ソフトを解除するなどしてフリーエリアを広げる。

(ENGLISH):Interface Device is not ready.
(日本語):未接続のインターフェースを指定しました.
原因と対処:'-I'オプションにおいて、利用できないインタフェイスは指定しても意味がない。

(ENGLISH):Interface Device sequence error.
(日本語):同一インターフェースを複数回指定しました.
原因と対処:-Iオプションおいて、1つのインタフェイスは1度しか指定できない。

(ENGLISH):ZMUSIC SYSTEM ver.3.0 runs on Human68k ver.3.00 and over.
(日本語):ZMUSICシステム ver.3.0はHuman68k ver.3.00以上で動作します.

(ENGLISH):ZMSC3.X cannot be registered from CONFIG.SYS.
(日本語):ZMSC3.XはCONFIG.SYSから登録することは出来ません.
原因と対処:ver.2.0以前はCONFIG.SYSから組み込めたがver.3.0以上はCOMMAND.X上からしか組み込めない。

(ENGLISH):ZMUSIC PERFORMANCE MANAGER is not kept in the system.
(日本語):ZMUSIC演奏マネージャは常駐していません.

(ENGLISH):ZMUSIC PERFORMANCE MANAGER is unable to release.
(日本語):ZMUSIC演奏マネージャは常駐解除できません.

(ENGLISH):ZMUSIC COMPILER is occupied by some other application.
(日本語):ZMUSICコンパイラが他のアプリケーションに占有されています.
原因と対処:Z-MUSICコンパイラ(ZMC.X)を占有しているアプリケーションを解除してからZMSC3.Xを解除する。

(ENGLISH):ZMUSIC PERFORMANCE MANAGER is occupied by some other applications.
(日本語):ZMUSIC演奏マネージャが他のアプリケーションに占有されています.
原因と対処:Z-MUSIC演奏マネージャ(ZMSC3.X)を占有しているアプリケーションを解除してからZMSC3.Xを解除する。

(ENGLISH):Illegal version number. Unable to release.
(日本語):バージョンが異なるため常駐解除はできません.

11.4. ZMC.Xのエラー

(ENGLISH):ZMUSIC MML COMPILER is not kept in the system.
(日本語):ZMUSIC MML コンパイラは常駐していません

(ENGLISH):ZMUSIC MML COMPILER is unable to release.
(日本語):ZMUSIC MML コンパイラの常駐解除はできません

(ENGLISH):ZMUSIC MML COMPILER is occupied by some other application.
(日本語):ZMUSIC MML コンパイラは他のアプリケーションに占有されています
原因と対処:Z-MUSIC MML コンパイラ(ZMC.X)を占有しているアプリケーションを先に常駐解除する。

(ENGLISH):Illegal version number. Unable to release.
(日本語):バージョンが異なるため解除は出来ません

(ENGLISH):ZMUSIC PERFORMANCE MANAGER has to be included previously.
(日本語):ZMUSIC演奏マネージャが先に常駐していなければなりません
原因と対処:ZMC.X単体の常駐はできない。

(ENGLISH):Fail in registering ZMUSIC MML COMPILER.
(日本語):ZMUSIC MML コンパイラの登録に失敗しました

(ENGLISH):ZMC.X cannot be registered from CONFIG.SYS.
(日本語):ZMC.XはCONFIG.SYSから登録することは出来ません
原因と対処:ver.2.0以前ではCONFIG.SYSからZ-MUSICを組み込めたが、ver.3.0以上では処理系の都合上、COMMAND.X上からしか組み込むことができない。

11.5. ZPCNV3.Rのエラー

(ENGLISH):File open error. Does the FILENAME1 file surely exist?
(日本語):ファイルが開けませんでした。FILENAME1が存在するか確認してください。

(ENGLISH):File open error. Check the legitimacy of FILENAME2.
(日本語):ファイルが開けませんでした。FILENAME2が適正なファイル名か確認してください。

(ENGLISH):File read error. Check the legitimacy of FILENAME1.
(日本語):読み込みエラーです。FILENAME1が正常か確認してください。

(ENGLISH):File write error. Check the legitimacy of FILENAME2.
(日本語):書き込みエラーです。FILENAME2が正常か確認してください。

(ENGLISH):Device full.
(日本語):ディスク容量が不足しています。
原因と対処:ディスクを増設/交換するか、不要なファイルを削除してフリーエリアを広げる。

(ENGLISH):Out of memory.
(日本語):メモリが不足しています。
原因と対処:メモリを増設するか、不要な常駐ソフトを解除してフリーエリアを広げる。

(ENGLISH):Illegal File size.
(日本語):ファイルサイズが異常です。
原因と対処:ファイルサイズがメインメモリより大きくないか、ファイルサイズが0になっていないか確認する。

11.6. ZP.Rのエラー

(ENGLISH):ZMUSIC Interrapt Service has already been used by other applications.
(日本語):ZMUSIC割り込みサービスは既に他のプログラムが利用中です
原因と対処:先にZ-MUSIC割り込みサービスを使用しているソフトの常駐を解除する必要がある。

(ENGLISH):MIDI receiver program dropped data.
(日本語):MIDI受信プログラムがデータを取りこぼしました
原因と対処:CPUの負荷になるソフトをすべて常駐解除してもう一度試してみる。

(ENGLISH):ZMUSIC refused ZP3.R registration.
(日本語):ZMUSICにZP3.Rの登録を拒否されました
原因と対処:すでに同種のプレイヤーソフトウェアが常駐している可能性がある。

(ENGLISH):ZMUSIC is not included.
(日本語):ZMUSICが組み込まれていません
原因と対処:ZP3.R単体での常駐は行えない。

(ENGLISH):Illegal file size.
(日本語):ファイルサイズが異常です
原因と対処:ファイルサイズがメインメモリより大きくないか、ファイルサイズが0になっていないか確認する。

(ENGLISH):Blank time too long.
(日本語):ブランクタイムが長すぎます
原因と対処:ブランクタイムの有効範囲は0-65535。

(ENGLISH):Illegal parameter error.
(日本語):値が規定外です

(ENGLISH):Illegal channel ID error.
(日本語):チャンネルIDが規定外です

(ENGLISH):Illegal interface ID error.
(日本語):インターフェースIDが規定外です

(ENGLISH):Too many track numbers error.
(日本語):トラック番号が多すぎます
原因と対処:ZP3.Rから一度に指定できるトラック数は1〜128までに限られる。

(ENGLISH):Out of memory.
(日本語):メモリが不足しています
原因と対処:メモリを増設するか、不要な常駐ソフトを解除してフリーエリアを広げる

(ENGLISH):File read error.
(日本語):ファイルの読み込みに失敗しました
原因と対処:ディスクドライブやファイルが破壊されていないか確認してみる。

(ENGLISH):File not found.
(日本語):ファイルが見つかりませんでした

(ENGLISH):Too many filenames are written in index-file.
(日本語):インデックスファイル中に記述されたファイル名が多すぎます
原因と対処:インデックスファイル中には最大128ファイルを記述することができる。

(ENGLISH):Unidentified file.
(日本語):認識できないファイルです
原因と対処:そのファイルが本当にZ-MUSIC システムで扱えるものかを確認してみる。

(ENGLISH):This is not a performance data.
(日本語):これは演奏データではありません

(ENGLISH):Unidentified ZDF file.
(日本語):認識できないZDFファイルです
原因と対処:Z-MUSIC ver.3.0で扱えるZDFはZMS,ZMD,ZPD,MDDいずれかを圧縮したものに限られる。

(ENGLISH):This data contains an unsupportable external file reference.
(日本語):未対応の外部ファイル参照を含んでいます
原因と対処:ジュークボックスで連続演奏する曲データには、(AD)PCM定義ファイル(CNFファイル)のような実行処理にZMC.Xを必要とするものを含んでいてはならない。

(ENGLISH):Version number mismatch.
(日本語):バージョン番号が不一致です

(ENGLISH):MIDI is Unable to use.
(日本語):MIDIは使用できません
原因と対処:MIDIボードを装着するかRS-232C MIDIを使用する設定でZMSC3.Xを組み込んでみる。

(ENGLISH):ZP3.R is not kept in your system.
(日本語):ZP3.Rは常駐していません

(ENGLISH):ZP3.R has already been kept.
(日本語):ZP3.Rは既に常駐しています

(ENGLISH):LZZ.R couldn't be found.
(日本語):LZZ.Rが見つかりません
原因と対処:音楽演奏データ圧縮ツールLZZ.Rがパスの通ったディレクトリに存在するかどうか確認してみる。

(ENGLISH):LZZ.R couldn't be used.
(日本語):LZZ.Rを利用することができませんでした

(ENGLISH):Error in LZZ.R.
(日本語):LZZ.R内でエラーが発生しました

(ENGLISH):File write error.
(日本語):ファイルの書き出しに失敗しました
原因と対処:ディスクの空き容量が十分か、ディスクが破壊されていないかを確認する。

(ENGLISH):Illegal fader speed.
(日本語):フェーダースピードの値が異常です
原因と対処:フェーダースピードの有効範囲は1-255。

(ENGLISH):Temporary file was broken.
(日本語):テンポラリファイルが異常です
原因と対処:テンポラリドライブが正常に動作するかを確認する。

(ENGLISH):Fail in execution of ZMUSIC MML COMPILER.
(日本語):ZMUSIC MMLコンパイラの起動に失敗しました

(ENGLISH):Error in execution of ZMUSIC MML COMPILER.
(日本語):ZMUSIC MMLコンパイラでエラーが発生しました

(ENGLISH):Fail in a calculation of total step time.
(日本語):トータルステップタイムの計算に失敗しました

(ENGLISH):No data was received.
(日本語):データは受信されませんでした
原因と対処:MIDIケーブルが正しく接続されているか、断線していないかを確認してみる。

MEASURE 12

ZMD／ZPDフォーマット

ここではZMD, ZPD、その他のZ-MUSIC ver.3.0で扱うファイルのフォーマットについて解説します。

１２．１．　ZMDファイルのヘッダブロックの構成

[注意]
文中の(.b)(.w)(.l)および(ds.b)(ds.w)(ds.l)などはデータ長を表している。
(ds.b)/(.b)　バイトサイズ
(ds.w)/(.w)　ワードサイズ(２バイト)
(ds.l)/(.l)　ロングワードサイズ(４バイト)

ZMDは、
$1a,'ZmuSiC',ZMD version ID($30)
のヘッダIDを先頭に以下のようなヘッダ構成を取る。

各ヘッダ項目は、各情報/データ領域までのオフセットを格納するもの(z_comn_offset,z_trk_offsetなど)と項目データそのものを格納するもの(z_total_count,z_tempoなど)の2タイプに分けられる。

演奏データなどを一切含んでいない「空」のZMDであっても以下に示したヘッダブロックは必ず存在する。

z_zmusic_id:	ds.l	2	*$1a,'ZmuSiC',ZMD version ID($30)
z_comn_offset:	ds.l	1	*共通コマンドまでのオフセット(０:なし)
z_trk_offset:	ds.l	1	*トラック/チャンネル情報テーブルまでの *オフセット(0:なし)
z_ctrl_offset:	ds.l	1	*演奏制御コマンドまでのオフセット *(０で全トラック演奏)
z_zmd_size:	ds.l	1	*このZMDファイル全体のサイズ
z_song_offset:	ds.l	1	*歌詞データまでのオフセット
	ds.l	1	*Reserved
z_total_offset:	ds.l	1	*トータルステップタイムの格納アドレス
z_title_offset:	ds.l	1	*タイトル文字列までのオフセット *(または最初のコメント文)
z_total_count:	ds.l	1	*Total Step Time
z_play_time:	ds.l	1	*演奏時間(時間/分/秒)　[reserved]
z_meter:	ds.w	1	*拍子(n/m)
z_metronome:	ds.b	1	*メトロノーム速度
	ds.b	1	*Reserved
z_key:	ds.w	1	*調号(値が正=ｎ個の＃,負=ｎ個のｂ，０=Ｃ)， *（0で長調/-1で短調)
z_master_clock:	ds.w	1	*全音符のマスタークロック[192]
z_tempo:	ds.w	1	*初期テンポ[120]
	ds.w	1	*Reserved
z_cmn_flag:	ds.l	1	*共通コマンドの使用状態
z_ctrl_flag:	ds.l	1	*制御コマンドの使用状態
z_instrument:	ds.l	1	*使用楽器タイプ(d0:GM d1:GS d2:SC-88 d3:MT-32 d4:U220 d5:M1)
z_nof_fm_ch:	ds.b	1	*FM音源使用チャンネル数(0-8)
z_nof_ad_ch:	ds.b	1	*AD PCM(PCM)音源使用チャンネル数(0-16)
z_nof_md1_ch:	ds.b	1	*MIDI-1使用チャンネル数(0-16)
z_nof_md2_ch:	ds.b	1	*MIDI-2使用チャンネル数(0-16)
z_nof_md3_ch:	ds.b	1	*MIDI-3使用チャンネル数(0-16)
z_nof_md4_ch:	ds.b	1	*Reserved
	ds.w	1	*Reserved

なお、タイトル文字は、1行目は曲のタイトル。それ以降はコメント。また、曲名/作曲者/編曲者/データ作成者のような一般的な情報を複数行にわたって記述する場合は、以下の文字列で表される情報タイトルの後に記載するようにする(情報タイトルの日本語は全角、英字は半角英数大文字のみ、また ： は必須)。

[日本語]	[英語]
曲名:	TITLE:
作曲者:	COMPOSER:
編曲者:	ARRANGER:
作成者:	MANIPULATER:
作成日:	DATE:　(原曲の作曲年月日はCOPYRIGHT:)
録音場所	STUDIO:
使用機器:	INSTRUMENTS:
ジャンル:	CATEGORY:

例

```
.comment {
TITLE: I am a boy, therefore I have balls.
作曲者:西川善司
STUDIO:自宅
ジャンル:童謡
}
```

１２．２．　共通コマンドのZMD

ZMDに共通コマンドが存在しない場合は、ヘッダブロックの z_comn_offset の内容を０とする。存在する場合はいちばん初めの共通コマンドZMDの存在位置までのオフセットアドレスを格納する。

共通コマンドZMDには以下のようなものがある。

> **オフセット値について**
>
> Z-MUSIC用のデータで扱うオフセット値はオフセット値の格納してある次のアドレスからのオフセットです。いまたとえば、$A000番地にワードサイズのオフセット$0E00が格納されていたとするとポイントするアドレスは$A002＋$0E00＝$AE02となります。

●CMN_END　　共通コマンドZMDの終了

$ff(.b)

●CMN_INIT　　初期化

$00(.b)　　ZMD
$00(.b)　　将来の拡張用
合計2bytes

●CMN_READ_SUB　　サブファイルの読み込みと実行

$04(.b)　　ZMD
Filenameファイル名文字列(Endcode=0(.b))
合計?bytes
または、
$04(.b)　　ZMD

0(.b)　文字列でなくオフセット値指定であることを示すコード
Offset(.l)　データが存在するアドレスまでのオフセット
Size(.l)
合計10bytes

●CMN_TEMPO　テンポ

$08(.b)　ZMD
Tempo(.w)　テンポ値:0-65535
合計3bytes

●CMN_MASTER_CLOCK 全音符クロック数登録

$0c(.b)　ZMD
Master Clock(.w)マスタークロック:0-32767
合計3bytes

●CMN_FM_TUNE_SETUP　FM音源チューニング

$10(.b)　ZMD
Data...(128bytes)　チューニング値(MML @Kレンジ):-128〜+127
合計129bytes

●CMN_PCM_TUNE_SETUP　(AD)PCM音源チューニング

$14(.b)　ZMD
Data...(128bytes)　チューニング値(MML @Kレンジ):-128〜+127
合計129bytes

●CMN_VSET:　FM音源音色設定

$18(.b)　ZMD
Tone Number(.w)　0-256*128
0(.b)　RESERVED
Tone Data(.b)　0)LFRQ
Tone Data(.b)　1)PMD
Tone Data(.b)　2)AMD
Tone Data(.b)　3)SYNC/OM/WF
Tone Data(.b)　4)PAN/AF
Tone Data(.b)　5)PMS/AMS
Tone Data(.b)　6)OP1:DT1/MUL
Tone Data(.b)　7)OP3:DT1/MUL
Tone Data(.b)　8)OP2:DT1/MUL
Tone Data(.b)　9)OP4:DT1/MUL
Tone Data(.b)　10)OP1:TL
Tone Data(.b)　11)OP3:TL
Tone Data(.b)　12)OP2:TL
Tone Data(.b)　13)OP4:TL
Tone Data(.b)　14)OP1:KS/AR
Tone Data(.b)　15)OP3:KS/AR
Tone Data(.b)　16)OP2:KS/AR
Tone Data(.b)　17)OP4:KS/AR
Tone Data(.b)　18)OP1:AME/1DR
Tone Data(.b)　19)OP3:AME/1DR
Tone Data(.b)　20)OP2:AME/1DR
Tone Data(.b)　21)OP4:AME/1DR
Tone Data(.b)　22)OP1:DT2/2DR
Tone Data(.b)　23)OP3:DT2/2DR
Tone Data(.b)　24)OP2:DT2/2DR
Tone Data(.b)　25)OP4:DT2/2DR
Tone Data(.b)　26)OP1:D1L/RR
Tone Data(.b)　27)OP3:D1L/RR
Tone Data(.b)　28)OP2:D1L/RR
Tone Data(.b)　29)OP4:D1L/RR
Tone Data(.b)　30)-45)NAME
合計50bytes

●CMN_WAVE_FORM　波形メモリ登録

$1c(.b)　ZMD
Wave Number(.w)　波形番号:8-255
Data Size(.l)　データサイズ:2〜　(必ず偶数)
Loop Type(.b)　波形タイプ:
(0:1Shot 1:Loop 2:Alternative-Loop)
Loop Start Offset(.l)　ループ開始オフセット:0〜　(必ず偶数)
(ループなしの場合は０)
Loop End Offset(.l)　ループ終端オフセット:0〜　(必ず偶数)
(オフセット値はループ終端データのあるオフセットアドレス+2)
(ループなしの場合は最終データの存在するオフセットアドレス+2)
Loop Time(.l)　ループ回数:(1〜)(０は無限回に相当)
Reserved(.l)　Reserved
String Length(.b)　コメント文字列長:0-255
String Data...　コメント文字列　(No Endcode)
(.even処理)
Wave Data(.w)　波形データ　(必ず偶数アドレスから...)
合計25〜?bytes

●CMN_REGISTER_PCM　(AD)PCM登録

$20(.b)　ZMD
Flag(.w)　フラグ
d15:PPCの有無(0:no 1:exist)
d0-d14:1-32767(次の共通コマンドまでのオフセット。0は15ビットでは表現できない場合を表すが通常は無視してよい。
Note/Timbre No.(.w)　Note Number(0-32767)/
Timbre Number($8000+0-32767)
Regtype(.b)　登録タイプ
(-1:AD PCM　0:V2互換　1:16bitPCM　2:8bitPCM)
Original Key Code(.b)　オリジナルキー
(0-127:Key Code　128:No Original Key)
Reserved(.w)　リザーブ
Tone Name(.b)...0(.b)　Tone Name(最大32バイト) (Endcode=0(.b))
Filename(.b)...0(.b)　ファイルネーム文字列　(Endcode=0(.b))
(Filename).l=$00xxxxxxの場合はすでに登録されているトーン/音色データをコピー流用する指定となる。
(Filename).l=$01xxxxxxの場合はすでに登録されているトーン/音色データを流用する。
(この場合、.LOOP以外のPPCは利用不可)
PPC(?bytes)　PCM PROCESSING COMMAND(PPC)...
PPCについては後述
合計21-?bytes

●CMN_ERASE_PCM　(AD)PCM登録取消

$24(.b)　ZMD
Note/Timbre No.(.w)　Note Number(0-32767)/
Timbre Number($8000+0-32767)
合計3bytes

●CMN_BLOCK_PCM　ZPD読み込み

$28(.b)　ZMD
Filename(.b)...0(.b)　ファイルネーム文字列(Endcode=0(.b))
合計3〜?bytes
または、
$28(.b)　ZMD
ZPD.ID(.b)　ZPD ID(0:V1〜2 1:V3 2:PDX)
Offset(.l)　データが存在するアドレスまでのオフセット
合計6bytes

●CMN_CRNT_MIDI_IN　カレントMIDI-IN設定

$2c(.b)　ZMD
MIDI I/F Number(.b)　インタフェイス番号:0-3
合計2bytes

●CMN_CRNT_MIDI_OUT　カレントMIDI-OUT設定

$30(.b)　ZMD
MIDI I/F Number(.b)　インタフェイス番号:0-3
合計2bytes

●CMN_MIDI_TRANSMISSION　MIDIデータ転送

$34(.b)　ZMD
MIDI I/F(.b)　インタフェイス番号:0-3/
(-1:カレントMIDI-OUT)
String Length(.b)　文字列長:0-255

String Data...	コメント文字列(No Endcode)
Transmit Data Length(.l)	転送データ長
Data....	転送データ
合計8-?bytes	

●CMN_MIDI_DUMP　　スタンダードMIDIファイルの送信

$38(.b)	ZMD
MIDI I/F(.b)	インタフェイス番号:0-3/ (-1:カレントMIDI-OUT)
Filename(.b)...0(.b)	ファイルネーム文字列(Endcode=0(.b))
合計4-?bytes	
または、	
$38(.b)	ZMD
MIDI I/F(.b)	インタフェイス番号:0-3/ (-1:カレントMIDI-OUT)
0(.b)	文字列でなくオフセット値指定であることを示すコード
Offset(.l)	データが存在するアドレスまでのオフセット
合計7bytes	

●CMN_COMMENT　　コメント

$40(.b)	ZMD
String(.b)...0(.b)	文字列(Endcode=0(.b))
合計3-?bytes	

●CMN_PRINT　　文字列表示

$44(.b)	ZMD
String(.b)...0(.b)	文字列(Endcode=0(.b))
合計3-?bytes	

●CMN_DUMMY　　ダミー

$48(.b)	ZMD
合計1byte	

●CMN_HALT　　停止

$4c(.b)	ZMD
Interval(.l)	待ち時間=1/60秒×Interval(.l)
合計5bytes	

１２．３．　PCM PROCESSING COMMANDのZMD

共通コマンドZMD「CMN_REGISTER_PCM」はPCMの加工をPCM PROCESSING COMMAND(PPC)を用いてver.2.0以前の(AD)PCM加工処理よりも柔軟な加工が行えるようになっている。以下に示すのはPPCのZMDで、共通コマンドZMD「CMN_REGISTER_PCM」中に用いて初めて機能し、これ単体で用いられることはない(用いることはできない)。

●PPC_END　　PPC終了

$00(.w)	PPC END

●PPC_PITCH　　音程変更

$02(.w)	PPC
Offset Count(.l)	加工開始オフセット:値は物理サイズではなくてデータカウント。
Processing Count(.l)	加工サイズ:値は物理サイズではなくてデータカウント。０は加工開始オフセット以降すべてを加工範囲とする。
Source Frq(.w)	変換元周波数:1-65535
Dest.　Frq(.w)	変換後周波数:1-65535
合計13bytes	

※データカウントは登録タイプによってその次元は変化する
・登録タイプがAD PCMのときデータ1カウントは物理量1バイトに相当する
・登録タイプが16bitPCMのときにはデータ1カウントは物理量2バイトに相当する。その他のPPCについても同様

●PPC_VOLUME　　音量変更

$04(.w)	PPC
Offset Count(.l)	加工開始オフセット:値は物理サイズではなくてデータカウント。
Processing Count(.l)	加工サイズ:値は物理サイズではなくてデータカウント。０は加工開始オフセット以降すべてを加工範囲とする。
Volume(.w)	音量:0-65535
合計11bytes	

●PPC_MIX合成

$06(.w)	PPC
$0000(.w)	固定値
Mix Source(.w)	合成元:(Tone:0-32767 Timbre:$8000+(0-32767))
Offset Count(.l)	合成開始オフセット:値は物理サイズではなくてデータカウント(負の値も可能) 負の場合は絶対値カウント分、合成先データの前側に合成する指定となる。
PPC...	PPCをさらにここに含め合成元のデータに対して加工処理を施すことができる。 PPC_ENDで合成元のデータに対するPPC列の終端を表す。
0(.b)	PPC_MIX終了
合計10-?bytes	
または、	
$06(.w)	PPC
Filename...0(.b)	ファイルネーム文字列(0=Endcode(.b))
Offset Count(.l)	合成開始オフセット:値は物理サイズではなくてデータカウント(負の値も可能) 負の場合は絶対値カウント分、合成先データの前側に合成する指定となる。
PPC...	PPCをさらにここに含め合成元のデータに対して加工処理を施すことができる。 PPC_ENDで合成元のデータに対するPPC列の終端を表す。
0(.b)	PPC_MIX終了
合計8-?bytes	

●PPC_TRUNCATE　　切り出し

$08(.w)	PPC
Offset Count(.l)	切り出し開始オフセット:値は物理サイズではなくてデータカウント。
Truncate Count(.l)	切り出しサイズ:値は物理サイズではなくてデータカウント。０はオフセット以降すべてを加工範囲とする。
合計9bytes	

●PPC_REVERSE　　反転

$0a(.w)	PPC
Offset Count(.l)	反転開始オフセット:値は物理サイズではなくてデータカウント。
Reverse Count(.l)	反転サイズ:値は物理サイズではなくてデータカウント。０はオフセット以降すべてを加工範囲とする。
合計9bytes	

●PPC_FADE　　フェーダー操作

$0c(.w)	PPC
Offset Count(.l)	加工開始オフセット:値は物理サイズではなくてデータカウント。
Fade Count(.l)	加工サイズ:値は物理サイズではなくてデータカウント。０はオフセット以降すべてを加工範囲とする。
Start Level(.w)	開始音量:0-128-65535(128は原音レベル)
End Level(.w)	終了音量:0-128-65535(128は原音レベル)
合計13bytes	

●PPC_BEND　　ピッチベンド

$0e(.w)	PPC
Offset Count(.l)	加工開始オフセット:値は物理サイズではなくてデータカウント
Bend Count(.l)	加工サイズ:値は物理サイズではなくてデータカウント。０はオフセット以降すべてを加工範囲とする。

Start Frq(.w)　開始周波数:1-65535[Hz]
End Frq(.w)　終了周波数:1-65535[Hz]
合計13bytes

●PPC_LOOP　ループポイント設定

$10(.w)　PPC
omt(.b)　省略フラグ(bit=1で対応パラメータが後続する)
　d0:loop start point
　d1:loop end point
Loop Type(.b)　ループタイプ:0=FORWARD　(現在は0のみ)
Loop Time(.l)　ループ回数:1～　(0は無限回に相当)
Loop Start(.l)　ループ開始オフセット:値は物理位置ではなくてデータカウント(omt.d0=1のときだけ設定)。
Loop End(.l)　ループ終了オフセット:値は物理位置ではなくてデータカウント(omt.d1=1のときだけ設定)。
合計6-14bytes

●PPC_CONNECT　接合

$12(.w)　PPC
$0000(.w)　固定値
Connect Source(.w)　接合元:(Tone:0-32767 Timbre:$8000+(0-32767))
Offset Count(.l)　接合開始オフセット:値は物理サイズではなくてデータカウント(負の値も可能)。
　負の場合は絶対値カウント分、接合先データの前側に接合する指定となる。
PPC...　PPCをさらにここに含め接合元のデータに対して加工処理を施すことができる。
　PPC_ENDで接合元のデータに対するPPC列の終端を表す。
0(.b)　PPC_CONNECT終了
合計11-?bytes
　または、
$12(.w)　PPC
filename...0(.b)　ファイルネーム文字列(Endcode=0(.b))
Offset Count(.l)　接合開始オフセット:値は物理サイズではなくてデータカウント(負の値も可能)。
　負の場合は絶対値カウント分、接合先データの前側に接合する指定となる。
PPC...　PPCをさらにここに含め接合元のデータに対して加工処理を施すことができる。
　PPC_ENDで接合元のデータに対するPPC列の終端を表す。
0(.b)　PPC_CONNECT終了
合計9-?bytes

●PPC_DELETE　削除

$14(.w)　PPC
Offset Count(.l)　削除開始オフセット:値は物理サイズではなくてデータカウント。
Truncate Count(.l)　削除サイズ:値は物理サイズではなくてデータカウント。0はオフセット以降すべてを削除範囲とする。
合計9bytes

●PPC_DISTORTION　ディストーション

$16(.w)　PPC
Offset Count(.l)　加工開始オフセット:値は物理サイズではなくてデータカウント。
Processing Count(.l)　加工サイズ:値は物理サイズではなくてデータカウント。0はオフセット以降すべてを加工範囲とする。
Volume(.w)　音量倍率:0-65535
Cut Off Level(.w)　音量臨界値:0-32767(絶対値をこれ以下のレベルに丸める)
合計13bytes

●PPC_SMOOTH　平均化(スムーズ化)

$18(.w)　PPC
Offset Count(.l)　加工開始オフセット:値は物理サイズではなくてデータカウント。
Reverse Count(.l)　加工サイズ:値は物理サイズではなくてデータカウント。0はオフセット以降すべてを加工範囲とする。
Repeat(w)　処理繰り返し回数:0～65535
　(0は1回,65535は65536回)
合計11bytes

12.4.　制御コマンドのZMD

ZMDに制御コマンドが存在しない場合はヘッダブロックのz_ctrl_offsetの内容を0とする。存在する場合は一番初めの制御コマンドZMDの存在位置までのオフセットアドレスを格納する。
　制御コマンドZMDには以下のようなものがある。

●CTRL_END　制御コマンド終了

$ff(.b)　ZMD

●CTRL_PLAY　演奏開始

$00(.w)　ZMD
Track Number(.w)　トラック番号:0-65534
Track Number(.w)　トラック番号:0-65534
　:
　:
Track Number(.w)　トラック番号:0-65534
-1(.w)　CTRL_PLAY終了
合計4-?bytes

●CTRL_STOP　演奏停止

$04(.w)　ZMD
Track Number(.w)　トラック番号:0-65534
Track Number(.w)　トラック番号:0-65534
　:
　:
Track Number(.w)　トラック番号:0-65534
-1(.w)　CTRL_STOP終了
4-?bytes

●CTRL_CONT　演奏再開

$08(.w)　ZMD
Track Number(.w)　トラック番号:0-65534
Track Number(.w)　トラック番号:0-65534
　:
　:
Track Number(.w)　トラック番号:0-65534
-1(.w)　CTRL_STOP終了
合計4-?bytes

●CTRL_MFADER　マスターフェーダー

$0c(.w)　ZMD
Device-ID(.w)　音源ID:(FM=0,(AD)PCM=1,MIDI1=$8000,MIDI2=$8001,MIDI3=$8002,MIDI4=$8003,-1=全音源が操作対象となる)
omt(.b)　省略フラグ(bit=1で対応パラメータが後続する)
　d0:speed
　d1:Start Level
　d2:End Level
Speed(.w)　移動速度:0-65535
Start Level(.b)　開始レベル:0-128(128=原音レベル)
End Level(.b)　終了レベル:0-128(128=原音レベル)
合計5-8bytes

●CTRL_TFADER　トラックフェーダー

$10(.w)　ZMD
Track Number(.w)　トラック番号:0-65534,-1=全トラックが操作対象となる。
omt(.b)　省略フラグ(bit=1で対応パラメータが後続する)
　d0:Speed
　d1:Start Level
　d2:End Level
Speed(.w)　移動速度:0-65535(0=移動しない)

```
Start Level(.b)         開始レベル:0-128(128=原音レベル)
End Level(.b)           終了レベル:0-128(128=原音レベル)
合計5-8bytes
```

●CTRL_MASK　　マスク制御

```
$14(.w)                 ZMD
    {
        Track Number(.w) トラック番号:0-65534
        Mask Mode(.w)   マスクモード:0=MASK OFF,
                                1=REVERSE,
                                -1=MASK ON
    }×n
-1(.w)                  CTRL_MASK終了
合計8-?bytes
```

全トラックを操作対象とする場合は、

```
$14(.w)                 ZMD
                        -1(.w)  全トラックを操作対象とするコード
                        Mask Mode(.w)マスクモード:0=MASK OFF,
                                        1=REVERSE,
                                        -1=MASK ON
合計6bytes
```

１２．５．　トラックチャンネル情報テーブルの構成

演奏データが存在しない場合はヘッダブロックのz_trk_offsetの内容を０とする。存在する場合はトラックチャンネル情報テーブルの存在位置までのオフセットアドレスを格納する。

トラックチャンネル情報テーブルとは、演奏データまでのオフセットアドレス、そのトラックに割り当てられた初期音源種別などの情報を格納した領域のこと。

以下にそのトラックチャンネル情報テーブルの構成を示す。

```
ti_n_of_tracks: ds.w  1 *.w トラック数-1(0-65535)

ti_track_stat:  ds.b  1 *.b $00=performance,$80=非演奏trackマーク
                        *   (詳しくはZMID.MAC参照)
ti_track_mode:  ds.b  1 *.b $80=no key off(詳しくはZMID.MAC参照)
ti_trkfrq:      ds.b  1 *.b 割り込み比率(0:normal,1:2回に1回,
                        *                2:3回に１回,...)
                ds.b  1 *.b
ti_type:        ds.w  1 *.w デバイスタイプ(FM:0 AD PCM:1
                        *                 MIDI:$8000～$8003,
                        *                 PATTERN:$7fff,
                        *                 CURRENT-MIDI:$ffff)
ti_ch:          ds.w  1 *.w チャンネル(0-15)
ti_play_data:   ds.l  1 *.l 演奏データまでのオフセット
ti_extra_info:  ds.l  1 *.l トラック追加情報までのオフセット
                                              (０の場合はない)
```

以降、ti_track_stat～ti_extra_infoまでが総トラック数分続く

１２．６．　トラック追加情報

トラック追加情報とは演奏トラックひとつひとつに対して付加される情報で、トラックチャンネル情報テーブル中のti_extra_infoから参照されるもの。トラック追加情報がない場合はti_extra_infoには0を格納する。

以下にトラック追加情報の構成を示す。

```
tx_total_step:          ds.l  1 *.l  トラック総ステップタイム
tx_checksum:            ds.l  1 *.l  トラックチェックサム
tx_n_of_measure:        ds.l  1 *.l  総小節数
tx_comment_length:      ds.l  1 *.l  トラックコメント文字列長
                                                      (0:none)
```

以降、tx_comment_lengthで表される長さの文字列が続く。ただしEndcode=0(.b)などはない。

１２．７．　演奏トラックのZMD

演奏トラックには以下に示すZMDが格納される。演奏トラックが存在する場合、トラックチャンネル情報テーブルのti_play_dataにその位置までのオフセットアドレスが格納される。

[注意]

・(v)で表されるデータサイズは1バイト～２バイトの任意のサイズを取ることを意味する。

・[]で表される数値はデフォルトであることを意味する。

・「→XXXX」などは、そのパラメータがMEASURE13で解説されるZ-MUSIC ver.3.0の演奏トラックワークに格納されることを意味している。

■$00～$7f

●音符(MML C,D,E,F,A,G)

```
Note Number(.b) ノート番号:0-127(.b)                    →k_note
Step Time(.v)   ステップタイム:0-127(.b), 32768+(0-32767)(.w)
                                                        →p_step_time
Gate Time(.v)   ゲートタイム:$8000(.w)=tie,
                  0-127(.b) | 32768+(128-32767)(.w) →k_gate_time
Velocity(.b)    ベロシティ:0-127, 128=use default,   →p_velo
                129～191,192,193～255は-63～-1,0,+1～+63に対応する
合計4-6bytes
```

■$80～$8f

●休符(MML R)

```
$80(.b)         ZMD
Step Time(.v)   ステップタイム:0-127(.b) | 32768+(0-32767)(.w)
                                                        →p_step_time
Gate Time(.v)   ゲートタイム:$8000(.w)=tie
                              0-127(.b) | 32768+(128-32767)(.w)
                                                        →k_gate_time
合計3-5bytes
```

●ウェイト(MML @W)

```
$81(.b)         ZMD
Step Time(.v)   ステップタイム:0-127(.b) | 32768+(0-32767)(.w)
                                                        →p_step_time
合計2-3bytes
```

●トラックディレイ(MML [TRACK_DELAY])

```
$82(.b)         ZMD
Step Time(.v)   0-127(.b), 32768+(0-32767)(.w)          →p_step_time
合計2-3bytes
```

●MXDRV系音符

```
$83(.b)         ZMD
Note Number(.b) ノート番号:0-127(+$80ではtie指定となる)
                                                        →k_note
Step Time(.v)   ステップタイム:0-127(.b) | 32768+(0-32767)(.w)
                                                        →p_step_time
合計3-6bytes
```

●ポルタメント1(MML (～))

```
$84(.b)         ZMD
Note Number(.b) ノート番号:0-127                  →k_note
                  $80+(0-127)の場合はディレイあり
Dest.Note(.b)   ポルタメント先のノート番号:0-127
                  $80+(0-127)の場合はポルタメントタイムあり
Delay(.v)       ディレイ:0-127(.b) | 32768+(0-32767)(.w)
                                                        →p_port_dly
                 (ただしNote Number(.b)≧$80のときのみ)
Port.Time(.v)   ポルタメントタイム:0-127(.b)           →p_port_cnt
                 32768+(0-32767)(.w)
                                  (ただしDest.Note≧$80のときのみ)
Step Time(.v)   ステップタイム:1-127(.b) | 32768+(1-32767)(.w)
                                                        →p_step_time
Gate Time(.v)   ゲートタイム:$8000(.w)=tie            →k_gate_time
                 0-127(.b) | 32768+(128-32767)(.w)
Velocity(.b)    ベロシティ:0-127 | 128=use default,  →p_velo
                129～191,192,193～255は-63～-1,0,+1～+63に対応する
合計6-11bytes
```

●ポルタメント2(MML([～]))

$85(.b)　ZMD
Note Number(.b)　ノート番号:0-127　→k_note
$80+(0-127)の場合はディレイあり
Dest.Note(.b)　ポルタメント先のノート番号:0-127
$80+(0-127)の場合はポルタメントタイムあり
Delay(.v)　ディレイ:0-127(.b) | 32768+(0-32767)(.w)　→p_port_dly
(ただしNote Number(.b)≧$80のときのみ)
Port.Time(.v)　ポルタメントタイム:0-127(.b)　→p_port_cnt
32768+(0-32767)(.w)
(ただしDest.Note≧$80のときのみ)
Step Time(.v)　ステップタイム:1-127(.b) | 32768+(1-32767)(.w)　→p_step_time
Gate Time(.v)　ゲートタイム:$8000(.w)=tie　→k_gate_time
0-127(.b) | 32768+(128-32767)(.w)
Velocity(.b)　ベロシティ:0-127 | 128=use default,　→p_velo
129〜191,192,193〜255は-63〜-1,0,+1〜+63に対応する
合計6-11bytes

■$90〜$9f

●音量(MML V,@V)
$90(.b)　ZMD
Volume(.b)　音量値:0-127　128段階指定時　→p_vol
$80+(0〜16)　16段階指定時
合計2bytes

●相対音量(MML _‾)
$91(.b) ZMD
Volume(.b)　相対音量値:-128〜127
合計2bytes

●相対音量2
$92(.b)　ZMD
Volume(.b)　相対音量値:-128〜127　→p_vol
合計2bytes
($91との違い:$91は必ず絶対音量次元で相対音量を設定するが、$92は直前の音量コマンド(128/16段階)の次元で相対音量を設定する)

●ベロシティ(MML U/@U)
$93(.b)
Velocity(.b)　ベロシティ値:0-127　128段階指定時　→p_velo
$80+(0〜16)　16段階指定時
合計2bytes

●相対ベロシティ(MML U±/@U±)
$94(.b) ZMD
Velocity(.b)　相対ベロシティ値:-128〜127　→p_velo
合計2bytes

●ピッチモジュレーションモード(MML M)
$95(.b)　ZMD
pmod mode(.b)　モード値:-1=normal　0=FM　1=MIDI　→p_pmod_mode
合計2bytes

●ピッチモジュレーションスイッチ(MML @M)
$96(.b)　ZMD
Switch(.b)　スイッチ:[0]=off　$ff=1/8mode　→p_pmod_sw
$01=optional 8 point mode　$02=on by previous mode
合計2bytes

●ピッチモジュレーション同期設定(MML H)
$97(.b)　ZMD
Sync Mode(.b)　同期モード　→p_pmod_flg
d7:キーオン時に波形初期化　0:OFF [1]:ON
d6:タイ時に波形初期化　[0]:OFF 1:ON
d5:休符時に波形初期化　[0]:OFF 1:ON
d4:波形をキーオン毎に更新　[0]:NO　1:YES
d3:波形を休符毎に更新　[0]:NO　1:YES
Delay(.w)　同期ディレイ:1〜65535　→p_pmod_syncnt
(0の場合は無指定を意味する)
合計4bytes

●オートベンドスイッチ(MML [BEND.SWITCH])
$98(.b)　ZMD
Switch(.b)　スイッチ:[0]=off　1=on　→p_bend_sw
合計2bytes

●アフタータッチシーケンススイッチ(MML [AFTERTOUCH.SWITCH])
$99(.b)　ZMD
Switch(.b)　スイッチ:[0]=off　$ff=1/8mode　→p_aftc_sw
$01=optional 8 point mode
$02=on by previous mode
合計2bytes

●アフタータッチシーケンス同期設定(MML [AFTERTOUCH.SYNC])
$9a(.b)　ZMD
Sync Mode(.b)　同期モード　→p_aftc_flg
d7:キーオン時に波形初期化　0:OFF [1]:ON
d6:タイ時に波形初期化　[0]:OFF 1:ON
d5:休符時に波形初期化　[0]:OFF 1:ON
Delay(.w)　同期ディレイ:1〜65535　→p_aftc_syncnt
(0の場合は無指定を意味する)
合計4bytes

●エンハンスドベロシティシーケンススイッチ (MML [VELOCITY.SWITCH])
$9b(.b)　ZMD
Switch(.b)　スイッチ:[0]=off　→p_arcc_sw+p_vseq_param
$01=optional 8 point mode,
$02=on by previous mode
合計2bytes

●アゴーギクスイッチ(MML [AGOGIK.SWITCH])
$9c(.b)　ZMD
switch(.b)　スイッチ:[0]=off　→(agogik_work_ptr)+p_pmod_sw
$ff=1/8mode
$01=optional 8 point mode
$02=on by previous mode
合計2bytes

●アゴーギク同期設定(MML [AGOGIK.SYNC])
$9d(.b)　ZMD
Sync Mode(.b)　同期モード　→p_agogik_flg
d7:キーオン時に波形初期化　[0]:OFF 1:ON
d6:タイ時に波形初期化　[0]:OFF 1:ON
d5:休符時に波形初期化　[0]:OFF 1:ON
d4:波形をキーオン毎に更新　[0]:NO　1:YES
d3:波形を休符毎に更新　[0]:NO　1:YES
d2:小節線毎に更新　[0]:NO　1:YES
Delay(.w)　同期ディレイ:1〜65535　→p_agogik_count
(0の場合は無指定を意味する)
合計4bytes

●タイモード(MML @J)
$9e(.b)　ZMD
Tie Mode　タイモード:(0=normal 1=special)　→d3 of p_md_flg
合計2bytes

●PCM再生モード(MML [PCM_MODE])
$9f(.b)　ZMD
PCM mode　PCMモード:(0=tone mode　1=timbre mode)　→d2 of p_md_flg
合計2bytes

■$a0〜$af

●パン 絶対指定(MML P/@P)
$a0(.b)　ZMD
Pan(.b)　パンポット:(0-127:L〜M〜R,128:off)　→p_pan

合計2bytes

●パン 相対指定(MML P/@P)
$a1(.b)　ZMD
Pan(.b)　相対パンポット値:-128～+127　→p_pan
合計2bytes

●スロットマスク(MML [SLOT_SEPARRATION])
$a2(.b)　ZMD
Mode(.b)　モード:%0000000～%1111000　→p_om
d3:OP1,d4:OP2,d5:OP3,d6:OP4
bit=1:off,bit=0:on
合計2bytes

●ダンパーペダル(MML @D)
$a3(.b)　ZMD
Damper(.b)　ダンパー値:0-63=off, 64-127=on　→p_damper
合計2bytes

●ノンオフモード(MML @R)
$a4(.b)　ZMD
Mode(.b)　モード:$00=通常, $80=ノートオフなし →p_track_mode
合計2bytes

●ベンドレンジ変更(MML G)
$a5(.b)　ZMD
Range(.b)　レンジ値:0-127 [12]　→p_@b_range
合計2bytes

●周波数変更(MML @F)
$a6(.b)　ZMD
Frq(.b)　周波数番号:0-6 [4]　→p_frq
合計2bytes

●絶対チャンネルプレッシャー
$a8(.b)　ZMD
Pressure(.b)　チャンネルプレッシャー値:0-127
合計2bytes

●相対チャンネルプレッシャー
$a9(.b)　ZMD
Pressure(.b)　相対チャンネルプレッシャー値:-128～1+27
(相対演算はベロシティと行われるが演算結果は
p_veloには反映されない)
合計2bytes

●キートランスポーズ(MML K)
$ab(.b)　ZMD
Value(.b)　キートランスポーズ値:-128～+127
合計2bytes

●FM/PCMトラックボイスリザーブ(MML [VOICE_RESERVE])
$ac(.b)　ZMD
n of voices(.b) ボイスリザーブ値:0-255=1～256　→p_voice_rsv
合計2bytes

●ベロシティ同期設定(MML [VELOCITY.SYNC])
$ad(.b)　ZMD
Sync Mode(.b)　同期モード　→p_arcc_flg+p_vseq_param
d7:小節線同期　[0]:off 1:on
d6:ベロシティが指定されている　[0]:off 1:on
ケースでもダミーで割り当てるか
d5:休符割り当て　[0]:off 1:on
Delay(.w)　同期ディレイ:1～65535 →p_arcc_syncnt+p_vseq_param
(0の場合は無指定を意味する)
合計4bytes

■$b0～$bf

●ピッチモジュレーション波形セレクト(MML S)
$b0(.b)　ZMD
Wave Form(.w)　波形番号:0-7=プリセット波形　→p_pmod_wf
(0:鋸歯波 1:矩形波 2:三角波 3:ワンショット鋸歯波
4:ランダム波)
$8000+0～32767=ユーザー波形(波形メモリ)
合計3bytes

●アゴーギク波形セレクト(MML [AGOGIK.WAVEFORM])
$b1(.b)　ZMD
Wave Form(.w)　波形番号:0-7=プリセット波形
→(agogik_work_ptr)+p_pmod_wf
$8000+(0～32767)=ユーザー波形(波形メモリ)
合計3bytes

●V2コンパチ音長0ノート(発音のみ行う)
$b2(.b)　ZMD
Note(.b)　ノート番号:0-127
Velocity(.b)　ベロシティ:0-127, 128=use default,　→p_velo
129～191,192,193～255は-63～-1,0,+1～+63に対応する
合計3bytes

●ARCCスイッチ(MML @A)
$b3(.b)　ZMD
Arcc No(.b)　ARCC番号:(0-3)×2
Switch(.b)　スイッチ:[0]=off $ff=1/8mode
$01=optional 8 point mode
$02=on by previous mode
→p_arcc_sw+p_arcc_param+(0-3)*trwk_size
合計3bytes

●ARCCモード(MML M)
$b4(.b)　ZMD
Arcc No(.b)　ARCC番号:(0-3)×2
Arcc Mode(.b)　モード:$ff=normal $01=extended mode
→p_arcc_mode
合計3bytes

●ARCC同期設定(MML H)
$b5(.b)　ZMD
Arcc No(.b)　ARCC番号:(0-3)×2
Sync Mode(.b)　同期モード　→p_arcc_flg
d7:キーオン時に波形初期化　0:OFF [1]:ON
d6:タイ時に波形初期化　[0]:OFF 1:ON
d5:休符時に波形初期化　[0]:OFF 1:ON
d4:波形をキーオン毎に更新　[0]:NO 1:YES
d3:波形を休符毎に更新　[0]:NO 1:YES
Delay(.w)　同期カウント:1～65535　→p_arcc_syncnt
(0の場合は無指定を意味する)
合計5bytes

●絶対OPMレジスタ書き込み(MML [OPM])
$b6(.b)　ZMD
Reg.Number(.b)　レジスタ番号:
$80+04:NFRQ $80+05:NE $80+06:LFRQ $80+07:PMD
$80+08:AMD $80+09:WF $80+10:CON $80+11:FB
$80+12:PAN $80+13:AMS $80+14:PMS
または、
(0:OP1,1:OP3,2:OP2,3:OP4)*16
+(0:MUL,1:DT1,2:TL,3:AR,4:KS,5:1DR,6:AMS-EN,
7:2DR,8:DT2,9:RR,10:1DL)
Data(.b)データ:0-255
合計3bytes

●相対OPMレジスタ書き込み(MML [OPM])
$b7(.b)　ZMD
Reg.Number(.b)　レジスタ番号:

$80+04:NFRQ $80+05:NE $80+06:LFRQ $80+07:PMD
$80+08:AMD $80+09:WF $80+10:CON $80+11:FB
$80+12:PAN $80+13:AMS $80+14:PMS

または、
(0:OP1,1:OP3,2:OP2,3:OP4)*16
+(0:MUL,1:DT1,2:TL,3:AR,4:KS,5:1DR,6:AMS-EN,
7:2DR,8:DT2,9:RR,10:1DL)

Data(.b)データ:-128〜+127
合計3bytes

●絶対ディチューン@B range(MML @B)
$b8(.b) ZMD
Detune(.w) ディチューン:-8192〜8191 →p_detune
合計3bytes

●絶対ディチューン@K range(MML @K)
$b9(.b) ZMD
Detune(.w) ディチューン:-7680〜7680 →p_detune
合計3bytes

●相対ディチューン@B range(MML @B±)
$ba(.b) ZMD
Detune(.w) ディチューン:-8192〜+8191 →p_detune
合計3bytes

●相対ディチューン@K range(MML @K±)
$bb(.b) ZMD
Detune(.w) ディチューン:-7680〜7680 →p_detune
合計3bytes

●レジスタ書き込み(コントロールチェンジ設定)(MML Y)
$bc(.b) ZMD
Register(.b) レジスタ:0-255
Data(.b)データ:0-255
合計3bytes

●強制再演奏(MML Jn)
$bd(.b) ZMD
Track(.w) トラック番号:0-65534
合計3bytes

●トラック同期(MML Wn)
$be(.b) ZMD
Track(.w) トラック番号:0-65534
合計3bytes

●Q/@Qコマンド
$bf(.b) ZMD
Q/@Q value(.w) Q/@Q値:0-256(ゲートタイムは(step*Q)/256
で与えられる)
@Q:-1〜-32768(ゲートタイムはstep+@Qで与えられる)
→p_Q_gate
合計3bytes

■$c0〜$cf

●ポリフォニックプレッシャー(MML [POLYPHONIC PRESSURE])
$c0(.b) ZMD
Note Number(.b) ノート番号:0-127=プレッシャー値は絶対指定
128+(0-127)=プレッシャー値は相対指定
Pressure(.b) プレッシャー値:0-127(絶対指定)
-127〜+127(相対指定)
(相対演算はベロシティと行われるが演算結果はp_veloには反映されない)
合計3bytes

●絶対テンポ(タイマ値)(MML @T)
$c1(.b) ZMD
Timer Value(.w) タイマ値:0-65535 →timer_value,m_tmp_buf
合計3bytes

●相対テンポ(タイマ値)(MML @T)
$c2(.b) ZMD
Timer Value(.w) 相対タイマ値:-32768〜32767 →timer_value,m_tmp_buf
合計3bytes

●絶対テンポ(MML T)
$c3(.b) ZMD
Tempo Value(.w) テンポ:0-65535 →timer_value,m_tmp_buf
合計3bytes

●相対テンポ(MML T)
$c4(.b) ZMD
Tempo Value(.w) 相対テンポ値:-32768〜32767 →timer_value,m_tmp_buf
合計3bytes

●シーケンスコマンド
(MML [D.C.] [DO] [!] [@] [KEY] [METER])
$c5(.b) ZMD
func no.(.b) 0:D.C. 1:DO 2:! 3:@ 4:KEY 5:METER
n of Param(.b) 後続パラメータ数:0-255
Params(.b) パラメータ群(n of Params(.b)の数だけ続く)
合計3-?bytes

●音色バンク切り換え(MML I)
$c6(.b) ZMD
Bank MSB(.b) バンク上位バイト:0-127 (負値の場合は無指定)
Bank LSB(.b) バンク下位バイト:0-127 (負値の場合は無指定)
合計3bytes

●音色切り換え(MML @)
$c7(.b) ZMD
Timbre Number(.w) 音色番号:0-32767 →p_pgm
合計3bytes

●音色切り換え2(MML [TIMBRE])
$c8(.b) ZMD
Timbre Number(.w) 音色番号:0-32767 →p_pgm
合計3bytes

●ARCC波形セレクト(MML S)
$c9(.b) ZMD
Arcc No(.b) ARCC番号:(0-3)×2
Wave Form(.w) 波形番号:0-7=プリセット波形 →p_arcc_wf
(0:鋸歯波 1:矩形波 2:三角波 3:ワンショット鋸歯波
4:ランダム波)
$8000+0〜32767=ユーザー波形(波形メモリ)
合計4bytes

●楽器ID設定(MML @I)
$ca(.b) ZMD
Maker(.b) メーカーID:0-127
(負値の場合は無指定) →p_maker
Device(.b) デバイスID:0-127
(負値の場合は無指定) →p_device
Module(.b) モジュールID:0-127
(負値の場合は無指定) →p_module
合計4bytes

●小節ジャンプ(MML [JUMP])
$cb(.b) ZMD
Measure No.(.l) 小説番号:0〜4294967295
合計5bytes

●チャンネルアサイン(MML N,@N,[CH_ASSIGN],[ASSIGN])
$cc(.b) ZMD
Device-ID(.w) 音源ID:(FM=0,(AD)PCM=1,MIDI1=$8000,MIDI2=$8001,
MIDI3=$8002,MIDI4=$8003) →p_type
Channel(.w) チャンネル番号:0-32767 →p_ch
合計5bytes

●リピート開始(MML |:)

$cd(.b) ZMD
Repeat Count(.w) 繰り返し回数:(0-[1]-65534:1-[2]-65535)
Repeat Work(.w) $0000.w ワーク
合計5bytes

●リピート終端(MML :|)

$ce(.b) ZMD
Offset(.l) リピート開始ZMD($cd)のRepeat Count(.w)までのオフセット値
合計5bytes

●NRPN(MML @Y)

$cf(.b) ZMD
Address H(.b) アドレス上位:0-127
Address L(.b) アドレス下位:0-127
Data H(.b) データ上位:0-127(128以上の値を設定した場合は省略とみなされる)
Data L(.b) データ下位:0-127(128以上の値を設定した場合は省略とみなされる)
合計5bytes0

■$d0〜$df

●SEGNO(MML [SEGNO])

$d0(.b) ZMD
Offset(.l) [D.S.]のD.S.Flag(.b)までのオフセットアドレス(0の場合はコマンド無視)
合計5bytes

●CODA(MML [CODA])

$d1(.b) ZMD
Offset(.l) [TOCODA]のCoda Flag(.b)までのオフセットアドレス
合計5bytes

●スキップZMD

$d2(.b) ZMD
Mode(.b) モード:0=offset 1=absolute
Offset(.l) Mode(.b)=0の場合は相対アドレスジャンプ
Mode(.b)=1の場合は絶対アドレスに相当する
合計6bytes

●D.S.(MML [D.S.])

$d3(.b) ZMD
D.S.Flag(.b) D.S.フラグ
d7:segno設定があったか(1:done),
d6:d.s.処理をしたか(1:done)
Offset(.l) [SEGNO]のZMDの次のZMDのアドレスまでのオフセット
→D.S.処理をしたらp_fine_flgをセット
合計6bytes

●TO CODA(MML [TOCODA])

$d4(.b) ZMD
Coda Flag(.b) CODAフラグ
d7:coda設定があったか(1:done)
d6:tocoda処理をしたか(1:done)
Offset(.l) [CODA]のZMDの次のZMDのアドレスまでのオフセット(0の場合はコマンド無視)
合計6bytes

●GOSUB(MML [GOSUB])

$d5(.b) ZMD
Track No.(.w) トラック番号:0-65534(ただし65535=パターントラック)
Offset(.l) 飛び先オフセットアドレス
合計7bytes

●チャンネルフェーダー(MML [CH_FADER])

$d6(.b) ZMD
Device-ID(.w) 音源ID:(FM=0,(AD)PCM=1,MIDI1=$8000,MIDI2=$8001,
MIDI3=$8002,MIDI4=$8003,カレントMIDI=$7ffd,
-1=全音源が操作対象となる)
Channel(.w) チャンネル番号:0-32767,-1=全チャンネル
omt(.b) 省略フラグ(bit=1で対応パラメータが後続する)
d0:Speed
d1:Start Level
d2:End Level
Speed(.w) 移動速度:0〜65535(0=移動しない,65535=最高速)
Start Level(.b) 開始レベル:0-128(128=原音レベル)
End Level(.b) 終了レベル:0-128(128=原音レベル)
合計6-9bytes
または、
$d6(.b) ZMD
$7fff(.w) 固定値
Track No.(.w) トラック番号:0-65534,-1=全トラック
omt(.b) 省略フラグ(bit=1で対応パラメータが後続する)
d0:Speed
d1:Start Level
d2:End Level
Speed(.w) 移動速度:0〜65535(0=移動しない,65535=最高速)
Start Level(.b) 開始レベル:0-128(128=原音レベル)
End Level(.b) 終了レベル:0-128(128=原音レベル)
合計6-9bytes
または、
$d6(.b) ZMD
$7ffe(.w) 固定値
Track ID.(.w) トラック識別子:0=現在のトラック -1=全トラック
omt(.b) 省略フラグ(bit=1で対応パラメータが後続する)
d0:Speed
d1:Start Level
d2:End Level
Speed(.w) 移動速度:0〜65535(0=移動しない,65535=最高速)
Start Level(.b) 開始レベル:0-128(128=原音レベル)
End Level(.b) 終了レベル:0-128(128=原音レベル)
合計6-9bytes

●マスターフェーダー(MML [MASTER_FADER])

$d7(.b) ZMD
Device-ID(.w) 音源ID:(FM=0,(AD)PCM=1,MIDI1=$8000,MIDI2=$8001,
MIDI3=$8002,MIDI4=$8003,
-1=全音源が操作対象となる)
omt(.b) 省略フラグ(bit=1で対応パラメータが後続する)
d0:Speed
d1:Start Level
d2:End Level
Speed(.w) 移動速度:0-65535(0=移動しない,65535=最高速)
Start Level(.b) 開始レベル:0-128(128=原音レベル)
End level(.b) 終了レベル:0-128(128=原音レベル)
合計4-7bytes

●リピートスキップ(MML |n)

$d8(.b) ZMD
Count(.w) 演奏許可繰り返し回数:0-65534(1-65535に対応)
Offset(.l) リピート開始のZMDのRepeat Work(.w)までのオフセットアドレス
Offset(.l) リピート終端のZMDの外までのオフセットアドレス
合計11bytes

●リピートスキップ2(MML |)

$d9(.b) ZMD
Offset(.l) リピート開始のZMDのRepeat Work(.w)までのオフセットアドレス
Offset(.l) 次のリピートスキップのZMDへの、または次のリピート終端のZMDまでのオフセットアドレス
合計9bytes

●ピッチモジュレーション振幅増加/減衰(MML [VIBRATO.DEEPEN])

$da(.b) ZMD
omt(.b) 省略フラグ(bit=1で対応パラメータが後続する)
d7:増減モード [0]:OFF 1:ON
d6:Speed 0:パラメータあり 1:なし

d5:Depth Accelarator　0:パラメータあり 1:なし
d4:Repeat Time　0:パラメータあり 1:なし
Speed(.v)　増減速度:0-127(.b), 32768+(0-32767)(.w)
Depth Acc(.w)　増減加速度:-32768〜32767
Repeat Time(.v)　増減繰り返し回数:0-127(.b), 32768+(0-32767)(.w)
0は無限回に相当する
合計5-8bytes

●ARCC振幅増加/減衰(MML [ARCCn.DEEPEN])

```
$db(.b)            ZMD
arcc no(.b)        (0-3)×2
omt(.b)            省略フラグ(bit=1で対応パラメータが後続する)
                   d7:増減モード            [0]:OFF 1:ON
                   d6:Speed                 0:パラメータあり 1:なし
                   d5:Depth Accelarator     0:パラメータあり 1:なし
                   d4:Repeat Time           0:パラメータあり 1:なし
Speed(.v)          増減速度:0-127(.b), 32768+(0-32767)(.w)
Depth Acc(.w)      増減加速度:-32768〜32767
Repeat Time(.v)    増減繰り返し回数:0-127(.b), 32768+(0-32767)(.w)
                                  0は無限回に相当する
合計6-9bytes
```

●ベロシティ振幅増加/減衰(MML [VELOCITY.DEEPEN])

```
$dc(.b)            ZMD
omt(.b)            省略フラグ(bit=1で対応パラメータが後続する)
                   d7:増減モード            [0]:OFF 1:ON
                   d6:Speed                 0:パラメータあり 1:なし
                   d5:Depth Accelarator     0:パラメータあり 1:なし
                   d4:Repeat Time           0:パラメータあり 1:なし
Speed(.v)          増減速度:0-127(.b), 32768+(0-32767)(.w)
Depth Acc(.b)      増減加速度:-128〜127
Repeat Time(.v)    増減繰り返し回数:0-127(.b), 32768+(0-32767)(.w)
                                  0は無限回に相当する
合計5-7bytes
```

●アゴーギク振幅増加/減衰(MML [AGOGIK.DEEPEN])

```
$dd(.b)            ZMD
omt(.b)            省略フラグ(bit=1で対応パラメータが後続する)
                   d7:増減モード            [0]:OFF 1:ON
                   d6:Speed          0:パラメータあり 1:なし
                   d5:Depth Accelarator     0:パラメータあり 1:なし
                   d4:Repeat Time           0:パラメータあり 1:なし
Speed(.v)          増減速度:0-127(.b), 32768+(0-32767)(.w)
Depth Acc(.w)      増減加速度:-32768〜32767
Repeat Time(.v)    増減繰り返し回数:0-127(.b), 32768+(0-32767)(.w)
                                  0は無限回に相当する
合計6-8bytes
```

●音色スプリット([TIMBRE_SPLIT],[PROGRAM_SPLIT])

```
$de(.b)                      ZMD
n of Param(.b)               スイッチフラグ/パラメータの個数
                             d7:スイッチ(0:OFF,1:ON)
                             d0-d6:パラメータの数:0-8
{
Range Start Note No.(.b)     スプリット開始ノート番号:0-127
Range End Note No.(.b)       スプリット終端ノート番号:0-127
Bank No.(.w)                 音色バンク:0-16383(MIDI)
                                        0-255(FM,AD PCM)
Timbre No.(.w)               音色番号:0-127 or 0-32767
}×n of Param(.b)             n of Param(.b)回パラメータ群が後
続                                 する
合計2-50bytes
```

●エンハンスドベロシティシーケンス波形セレクト (MML [VELOCITY.WAVEFORM])

```
$df(.b)                ZMD
omt(.b)                省略フラグ(bit=1で対応パラメータが後続する)
                        d0:Wave Form
                        d1:Origin Value
                        d1:Phase Mode
Wave Form(.w)          波形番号:0-7=プリセット波形
                        (0:鋸歯波 1:矩形波 2:三角波
                         3:ワンショット鋸歯波 4:ランダム波)
                         $8000+0〜32767=ユーザー波形(波形メモリ)
                                     →p_arcc_wf+p_vseq_param
Origin Value(.b)       波形の基準値:0-127
Phase Mode(.b)         位相モード:0,-1(0:正位相,-1:逆位相)
合計3〜6bytes
```

■$e0〜$ef

●オートベンド@B range(MML @B)

```
$e0(.b)                   ZMD
omt(.b)                   省略フラグ(bit=1で対応パラメータが後続する)
                           d7:Start Detune
                           d6:Destination Detune
                           d5:Delay
                           d4:Tail/Port.Time
                           (omt(.b)=0のときスイッチオフ)
Start detune(.w)          開始ディチューン:-32768〜32767 →p_detune
Destination detune(.w)    目的ディチューン:-32768〜32767→p_bend_dst
Delay(.w)                 ディレイ:-32768〜32767
                                          →p_bend_dly/p_port_cn
t
Tail/Port.Time(.w)        ベンド時間/持続時間:      →p_bend_tail
                           ベンド時間:1-32767
                           無指定:0
                           持続時間:-1〜-32768
合計2-10bytes
```

●オートベンド@K range(MML @K)

```
$e1(.b)                   ZMD
omt(.b)                   省略フラグ(bit=1で対応パラメータが後続する)
                           d7:Start Detune
                           d6:Destination Detune
                           d5:Delay
                           d4:Tail/Port.Time
                           (omt(.b)=0のときスイッチオフ)
Start detune(.w)          開始ディチューン:-32768〜32767 →p_detune
Destination detune(.w)    目的ディチューン:-32768〜32767→p_bend_dst
Delay(.w)                 ディレイ:-32768〜32767
                                         →p_bend_dly/p_port_cnt
Tail/Port.Time(.w)        ベンド時間/持続時間:        →p_bend_tail
                           ベンド時間:1-32767
                           無指定:0
                           持続時間:-1〜-32768
合計2-10bytes
```

●ピッチモジュレーション振幅(MML @M,[VIBRATO.DEPTH])

```
$e2(.b)           ZMD
Mode(.b)モード:0=無指定,1=OPTIONAL,-1=1/8 MODE,
                  2=以前のモードでON
omt(.b)           省略フラグ(bit=1で対応パラメータが後続する)
                  d0:1番目の振幅値に対応
                  d1:2番目の振幅値に対応
                  d2:3番目の振幅値に対応
                  d3:4番目の振幅値に対応
                  d4:5番目の振幅値に対応
                  d5:6番目の振幅値に対応
                  d6:7番目の振幅値に対応
                  d7:8番目の振幅値に対応
                  omt(.b)=0のときがスイッチオフの指定となる
Depth(.w)*(1-8)   振幅:-32768〜32767                →p_pmod_dpt_tbl
合計5-19bytes
```

●ピッチモジュレーションスピード(MML @S,[VIBRATO.SPEED])

```
$e3(.b)           ZMD
omt(.b)           省略フラグ(bit=1で対応パラメータが後続する)
                  d0:1番目の速度値に対応
                  d1:2番目の速度値に対応
```

　　　　　　　　d2:3番目の速度値に対応
　　　　　　　　d3:4番目の速度値に対応
　　　　　　　　d4:5番目の速度値に対応
　　　　　　　　d5:6番目の速度値に対応
　　　　　　　　d6:7番目の速度値に対応
　　　　　　　　d7:8番目の速度値に対応
　　　　　　　　omt(.b)=0はありえない
Speed(.w)*(1-8) 速度値:1-65535　→p_pmod_spd_tbl
　　　　　　　(0はエラー,値はMMLのパラメータで与えられたものを
　　　　　　　　2倍したもの)
合計4-18bytes

●ピッチモジュレーションディレイ(MML @H,[VIBRATO.DELAY])

$e4(.b)　　　　ZMD
omt(.w)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目のディレイ値に対応
　　　　　　　　d1:2番目のディレイ値に対応
　　　　　　　　d2:3番目のディレイ値に対応
　　　　　　　　d3:4番目のディレイ値に対応
　　　　　　　　d4:5番目のディレイ値に対応
　　　　　　　　d5:6番目のディレイ値に対応
　　　　　　　　d6:7番目のディレイ値に対応
　　　　　　　　d7:8番目のディレイ値に対応
　　　　　　　　d8:9番目のディレイ値に対応
　　　　　　　　omt(.b)=0はありえない
Delay(.w)*(1-9) ディレイ値:0-65535→p_pmod_1st_dly〜p_pmod_8st_tbl
合計5-21bytes

●ARCC振幅(MML @A,[ARCCn.LEVEL])

$e5(.b)　　　　ZMD
Arcc No(.b)　　ARCC番号:(0-3)×2
Mode(.b)　　　　モード:0=無指定,1=OPTIONAL,
　　　　　　　　　-1=1/8 MODE,2=以前のモードでON
omt(.b)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目の振幅値に対応
　　　　　　　　d1:2番目の振幅値に対応
　　　　　　　　d2:3番目の振幅値に対応
　　　　　　　　d3:4番目の振幅値に対応
　　　　　　　　d4:5番目の振幅値に対応
　　　　　　　　d5:6番目の振幅値に対応
　　　　　　　　d6:7番目の振幅値に対応
　　　　　　　　d7:8番目の振幅値に対応
　　　　　　　　omt(.b)=0のときがスイッチオフの指定となる
Depth(.b)*(1-8) 振幅値:-128〜127　→p_arcc_dpt_tbl
合計5-12bytes

●ARCCスピード(8point)(MML @S,[ARCCn,SPEED])

$e6(.b)　　　　ZMD
Arcc No(.b)　　ARCC番号:(0-3)×2
omt(.b)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目の速度値に対応
　　　　　　　　d1:2番目の速度値に対応
　　　　　　　　d2:3番目の速度値に対応
　　　　　　　　d3:4番目の速度値に対応
　　　　　　　　d4:5番目の速度値に対応
　　　　　　　　d5:6番目の速度値に対応
　　　　　　　　d6:7番目の速度値に対応
　　　　　　　　d7:8番目の速度値に対応
　　　　　　　　omt(.b)=0はありえない
Speed(.w)*(1-8) 速度値:1-65535　→p_arcc_spd_tbl
　　(0はエラー,値はMMLのパラメータで与えられたものを2倍したもの)
合計5-19bytes

●ARCCディレイ(MML @H,[ARCCn,DELAY])

$e7(.b)　　　　ZMD
Arcc No(.b)　　ARCC番号:(0-3)×2
omt(.w)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目のディレイ値に対応
　　　　　　　　d1:2番目のディレイ値に対応
　　　　　　　　d2:3番目のディレイ値に対応
　　　　　　　　d3:4番目のディレイ値に対応
　　　　　　　　d4:5番目のディレイ値に対応
　　　　　　　　d5:6番目のディレイ値に対応
　　　　　　　　d6:7番目のディレイ値に対応
　　　　　　　　d7:8番目のディレイ値に対応
　　　　　　　　d8:9番目のディレイ値に対応
　　　　　　　　omt(.b)=0はありえない
Delay(.w)*(1-9) ディレイ値:0-65535
　　　　　　　　　　　　→p_arcc_1st_dly〜p_arcc_8st_tbl
合計6-22bytes

●アフタータッチシーケンス
(1/8,optional 8point)(MML @Z,[AFTERTOUCH.LEVEL])

$e8(.b)　　　　ZMD
Mode(.b)　　　　モード:0=無指定,1=OPTIONAL,
　　　　　　　　　-1=1/8 MODE,2=以前のモードでON
omt(.b)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目のアフタータッチ値に対応
　　　　　　　　d1:2番目のアフタータッチ値に対応
　　　　　　　　d2:3番目のアフタータッチ値に対応
　　　　　　　　d3:4番目のアフタータッチ値に対応
　　　　　　　　d4:5番目のアフタータッチ値に対応
　　　　　　　　d5:6番目のアフタータッチ値に対応
　　　　　　　　d6:7番目のアフタータッチ値に対応
　　　　　　　　d7:8番目のアフタータッチ値に対応
　　　　　　　　omt(.b)=0のときがスイッチオフの指定となる
Relative(.b)　　相対値フラグ(bit=1で対応パラメータが相対指定である
　　　　　　　　ことを表す)
　　　　　　　　d0:1番目のアフタータッチ値に対応
　　　　　　　　d1:2番目のアフタータッチ値に対応
　　　　　　　　d2:3番目のアフタータッチ値に対応
　　　　　　　　d3:4番目のアフタータッチ値に対応
　　　　　　　　d4:5番目のアフタータッチ値に対応
　　　　　　　　d5:6番目のアフタータッチ値に対応
　　　　　　　　d6:7番目のアフタータッチ値に対応
　　　　　　　　d7:8番目のアフタータッチ値に対応
Level(.b)*(1-8) アフタータッチ値:0-127　→p_aftc_tbl
合計5-12bytes

●アフタータッチシーケンスディレイ(MML [AFTERTOUCH.DELAY])

$e9(.b)　　　　ZMD
omt(.w)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目のディレイ値に対応
　　　　　　　　d1:2番目のディレイ値に対応
　　　　　　　　d2:3番目のディレイ値に対応
　　　　　　　　d3:4番目のディレイ値に対応
　　　　　　　　d4:5番目のディレイ値に対応
　　　　　　　　d5:6番目のディレイ値に対応
　　　　　　　　d6:7番目のディレイ値に対応
　　　　　　　　d7:8番目のディレイ値に対応
　　　　　　　　d8:9番目のディレイ値に対応
　　　　　　　　omt(.b)=0はありえない
Delay(.w)*(1-9) ディレイ値:0-65535　→p_aftc_8st_tbl
合計5-21bytes

●エンハンスドベロシティシーケンス振幅(MML [VELOCITY.LEVEL])

$ea(.b)　　　　ZMD
Mode(.b)　　　　モード:0=無指定,1=OPTIONAL,2=以前のモードでON
omt(.b)　　　　省略フラグ(bit=1で対応パラメータが後続する)
　　　　　　　　d0:1番目の振幅値に対応
　　　　　　　　d1:2番目の振幅値に対応
　　　　　　　　d2:3番目の振幅値に対応
　　　　　　　　d3:4番目の振幅値に対応
　　　　　　　　d4:5番目の振幅値に対応
　　　　　　　　d5:6番目の振幅値に対応
　　　　　　　　d6:7番目の振幅値に対応
　　　　　　　　d7:8番目の振幅値に対応
　　　　　　　　omt(.b)=0のときがスイッチオフの指定となる
Level(.b)*(1-8) ベロシティ値:-128〜127　→p_arcc_dpt_tbl+p_vseq_pa
ram

合計4-11bytes

●エンハンスドベロシティシーケンススピード (MML [VELOCITY.SPEED])

```
$eb(.b)          ZMD
omt(.b)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:1番目の速度値に対応
                 d1:2番目の速度値に対応
                 d2:3番目の速度値に対応
                 d3:4番目の速度値に対応
                 d4:5番目の速度値に対応
                 d5:6番目の速度値に対応
                 d6:7番目の速度値に対応
                 d7:8番目の速度値に対応
                 omt(.b)=0はありえない
Speed(.w)*(1-8)  速度値:1-65535      →p_arcc_spd_tbl+p_vseq_param
                      (0はエラー,値はMMLのパラメータで与えられた
                       ものを2倍したもの)
```

合計4-18bytes

●エンハンスドベロシティシーケンスディレイ (MML [VELOCITY.DELAY])

```
$ec(.b)          ZMD
omt(.w)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:1番目のディレイ値に対応
                 d1:2番目のディレイ値に対応
                 d2:3番目のディレイ値に対応
                 d3:4番目のディレイ値に対応
                 d4:5番目のディレイ値に対応
                 d5:6番目のディレイ値に対応
                 d6:7番目のディレイ値に対応
                 d7:8番目のディレイ値に対応
                 d8:9番目のディレイ値に対応
                 omt(.b)=0はありえない
Delay(.w)*(1-9)  ディレイ値:0-65535    →p_arcc_8st_tbl+p_vseq_param
```

合計5-21bytes

●アゴーギク振幅(1/8,optional 8point)(MML [AGOGIK.LEVEL])

```
$ed(.b)          ZMD
Mode(.b)         モード:0=無指定,1=OPTIONAL,
                      -1=1/8 MODE,2=以前のモードでON
omt(.b)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:1番目の振幅値に対応
                 d1:2番目の振幅値に対応
                 d2:3番目の振幅値に対応
                 d3:4番目の振幅値に対応
                 d4:5番目の振幅値に対応
                 d5:6番目の振幅値に対応
                 d6:7番目の振幅値に対応
                 d7:8番目の振幅値に対応
                 omt(.b)=0のときがスイッチオフの指定となる
Depth(.w)*(1-8)  振幅:-32768～32767
                              →(agogik_work_ptr)+p_pmod_dpt_tbl
```

合計5-19bytes

●アゴーギクスピード(MML [AGOGIK.SPEED])

```
$ee(.b)          ZMD
omt(.b)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:1番目の速度値に対応
                 d1:2番目の速度値に対応
                 d2:3番目の速度値に対応
                 d3:4番目の速度値に対応
                 d4:5番目の速度値に対応
                 d5:6番目の速度値に対応
                 d6:7番目の速度値に対応
                 d7:8番目の速度値に対応
                 omt(.b)=0はありえない
Speed(.w)*(1-8)  速度値:1-65535  →(agogik_work_ptr)+p_pmod_spd_tbl
      (0はエラー,値はMMLのパラメータで与えられたものを2倍したもの)
```

合計4-18bytes

●アゴーギクディレイ(8point)(MML [AGOGIK.DELAY])

```
$ef(.b)          ZMD
omt(.w)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:1番目のディレイ値に対応
                 d1:2番目のディレイ値に対応
                 d2:3番目のディレイ値に対応
                 d3:4番目のディレイ値に対応
                 d4:5番目のディレイ値に対応
                 d5:6番目のディレイ値に対応
                 d6:7番目のディレイ値に対応
                 d7:8番目のディレイ値に対応
                 d8:9番目のディレイ値に対応
                 omt(.b)=0はありえない
Delay(.w)*(1-9)  ディレイ値:0-65535
                              →(agogik_work_ptr)+p_pmod_8st_tbl
```

合計5-21bytes

■$f0～$ff

●エフェクト設定(MML @E,[EFFECT])

・通常(GS音源など)

```
$f0(.b)          ZMD
omt(.b)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:Effect1
                 d1:Effect3
                 d2:Effect4
                 d3:Effect2
                 d4:Effect5
Effect1(.b)      エフェクト1値:0-127    (GS:リバーブ) →p_effect1
Effect3(.b)      エフェクト3値:0-127    (GS:コーラス) →p_effect3
Effect4(.b)      エフェクト4値:0-127    (GS:ディレイ) →p_effect4
Effect2(.b)      エフェクト2値:0-127                  →p_effect2
Effect5(.b)      エフェクト5値:0-127                  →p_effect5
```

合計3-8bytes

・MT-32系の場合(MT-32,CM-64など)

```
$f0(.b)          ZMD
omt(.b)          省略フラグ(bit=1で対応パラメータが後続する)
                 d0:PART No.
                 d1:Switch
PART No.(.b)     パート番号:0～7 or 8(リズム)
Switch(.b)       エフェクトスイッチ:0=OFF,1=ON
```

合計3-5bytes

●絶対ワーク直接書き換え(MML ?)

```
$f1(.b)               ZMD
Mode(.b)              下位4ビット:Data Offsetのデータサイズ
                                (0=1バイト 1=2バイト 3=4バイト)
                      上位4ビット:Addr.Offsetのデータサイズ
                                (0=1バイト 1=2バイト 3=4バイト)
Addr.Offset(b/w/l)    -2147483648～2147483647
Data (b/w/l)          0～4294967295
```

合計4-10bytes

●相対ワーク直接書き換え(MML ?)

```
$f2(.b)               ZMD
Mode(.b)              下位4ビット:Data Offsetのデータサイズ
                                (0=1バイト 1=2バイト 3=4バイト)
                      上位4ビット:Addr.Offsetのデータサイズ
                                (0=1バイト 1=2バイト 3=4バイト)
Addr.Offset(b/w/l)    -2147483648～2147483647
Data (b/w/l)          -2147483648～2147483647
```

合計4-10bytes

●エクスクルーシブ (MML X,[ROLAND_EXCLUSIVE],[YAMAHA_EXCLUSIVE])

```
$f3(.b)               ZMD
Exclusive Mode(.b)    0-255($ff:none $41:Roland $43:YAMAHA)
String Length(.b)     コメント文字列長:0-255
```

String Data...　　コメント文字列(No Endcode)
Data Length(.l)　　データ長:1〜4294967296
Data(.b)...　　データ列:0-127

(Exclusive Mode=$41,$43のとき、チェックサムが自動送信されるため、データ列にチェックサムを含む必要はない。)
合計9〜?bytes

●MIDIデータ転送(MML @X,[MIDI_DATA])

$f4(.b)　　ZMD
String Length(.b)　　コメント文字列長:0-255
String Data...　　コメント文字列(No Endcode)
Data Length(.l)　　データ長:1〜4294967296
Data(.b)...　　データ列:0-127
合計7〜?bytes

●ループ終端(MML [LOOP])

$f5(.b)　　ZMD
Offset(.l)*n　　オフセットアドレス値
(任意の個数続く,Offset(.l)=0が終端)
オフセットアドレスで指し示される任意の番地を0で初期化する。
合計5〜?bytes

●オートポルタメント(MML [AUTO_PORTAMENT])

$f6(.b)　　ZMD
Mode(.b)　　モード:0=OFF 1=ON -1=無指定　→p_port2_flg
omt(.b)　　省略フラグ(bit=1で対応パラメータが後続する)
d7:Delay
d6:Tail/Port.Time
Delay(.w)　　ディレイ値:-32768〜32767　→p_port2_dly
Tail/Port.Time(.w)　　ベンド時間/持続時間　→p_port2_tail
ベンド時間:1-32767
無指定:0
持続時間:-1〜-32768
合計3-7bytes

●ARCCアサイン(MML @C,[ARCCn.CONTROL])

$f7(.b)　　ZMD
Arcc No(.b)　　ARCC番号:(0-3)×2
omt(.b)　　省略フラグ(bit=1で対応パラメータが後続する)
d7:Ctrl Device
d6:Reset Value
d5:Origin Value
d4:Phase Mode
Ctrl Device(.b)　　MIDI:0-127 FM:$80=pan,$81=pms,$82=ams
$83=reserved,$84=amd,$85=pmd
$86=lfo spd,$87=noise spd
Reset Value(.b)　　MIDI:0-127 FM:0-255
Origin Value(.b)　　MIDI:0-127 FM:0-255
Phase Mode(.b)　　位相モード:0,-1(0:正位相,-1:逆位相)
合計4-6bytes

●イベント制御(MML [EVENT])

$f8(.b)　　ZMD
Size(.l)　　後続データのサイズ
(Category(.b)〜Data終端まで)
Category(.b)　　データの種類:(0=WORD,1=PICTURE,2=SOUND)
Object Class(.b)　　データの具体的な種類
Dummy(.w)　　将来拡張用
Data...　　データ
(データ構造はCategory,Classによって違う)
合計?bytes

・CategoryがWORDのとき

$f8(.b)　　ZMD
Size(.l)　　後続データのサイズ
(0のときはData部は0で終了するファイルネーム文字列
Category(.b)　　データの種類:0=WORD(歌詞)
Object Class(.b)　　データの具体的な種類
0=単なる文字列(1〜255:RESERVED)
Dummy(.w)　　将来拡張用
Data...　　データ列
Size(.l)=0の場合は0(.b)を終端コードに持つ文字列が格納される。
合計?bytes

・CategoryがPICTUREのとき

$f8(.b)　　ZMD
Size(.l)　　後続データのサイズ
0のときはData部は0で終了するファイルネーム文字列
Category(.b)　　データの種類:1=PICTURE(画像)
Object Class(.b)　　データの具体的な種類
0=PICファイル(1〜255:RESERVED)
Dummy(.w)　　将来拡張用
Data...　　データ列
Size(.l)=0の場合は0(.b)を終端コードに持つ文字列が格納される。
合計?bytes

・CategoryがSOUNDのとき

$f8(.b)　　ZMD
Size(.l)　　後続データのサイズ
0のときはData部は0で終了するファイルネーム文字列
Category(.b)　　データの種類:2=SOUND(音声)
Object Class(.b)　　後続データのサイズデータの具体的な種類
0=AD PCM(15.6kHz)(1〜255:RESERVED)
Dummy(.w)　　将来拡張用
Data...　　データ列
Size(.l)=0の場合は0(.b)を終端コードに持つ文字列が格納される。
合計?bytes
(カテゴリやクラスの新設追加は利用希望者がその都度Z-MUSICの制作者に申請する。連絡方法については巻末に記載。)

●リターン(GOSUB命令の次のアドレスへ帰還)

$f9(.b)　　ZMD
合計1byte

●ダミーZMD

$fa(.b)　　ZMD
合計1byte

●同期待ち(MML W,[SYNCHRONIZE])

$fb(.b)　　ZMD
合計1byte

●FINE(MML [FINE])

$fc(.b)　　ZMD
合計1byte

●強制キーオフ(MML `,[ALL_SOUND_OFF])

$fd(.b)　　ZMD
合計1byte

●小節線(MML [---])

$fe(.b)　　ZMD
合計1byte

●演奏データ終端

$ff(.b)　　ZMD
合計1byte

１２．８．　ZPDフォーマット

ZPDは、

$1a,'ZmaDPcM'

のヘッダIDを先頭に以下のようなヘッダ構成を取る。

各ヘッダ項目は、各情報/データ領域までのオフセットを格納するもの(zpd_offset,zpd_loop_startなど)と項目データそのものを格納するもの(zpd_note,zpd_typeなど)の2タイプに分けられる。

ひとつも(AD)PCMを含んでいない「空」のZPDであっても以下に示したヘッダブロックは必ず存在する。

```
zpd_header:      ds.b    8       *$1a,'ZmaDPcM'
                 dc.l    0       *Reserved
                 ds.l    1       *登録ノート数（０はなしを表す)

zpd_note:        ds.w    1       *登録番号
                                 *Tone:0-32767
                                 *Timbre:$8000+(0-32767)
zpd_type:        ds.b    1       *登録タイプ(-1:AD PCM  1:16bitPCM
                                 *              2:8bitPCM  0:No Data)
zpd_orig_key:    ds.b    1       *オリジナルキーのノート番号:0-127
zpd_attribute:   ds.b    1       *属性(d0=0:Normal,1:ﾎﾟｲﾝﾀｺﾋﾟｰ)
                 ds.b    1       *Reserved
zpd_offset:      ds.l    1       *オフセットアドレス値
zpd_size:        ds.l    1       *データ物理サイズ
zpd_loop_start:  ds.l    1       *ループ開始ポイント
zpd_loop_end:    ds.l    1       *ループ終了ポイント
zpd_loop_time:   ds.l    1       *ループ回数:0=∞,1～4294967295
                 ds.l    1       *Reserved
                 ds.l    1       *Reserved
zpd_tone_name:   ds.b    32      *トーンネーム(最大32文字)
```

・zpd_offset～zpd_loop_endは16bitPCM登録ケースは必ず偶数値となる。
・zpd_loop_start～zpd_loop_endはデータ先頭アドレスからの物理オフセットアドレス値を取る。

MEASURE 13

ワークエリアとワークビュアZSV.R

ここでは演奏トラックワークとその他のZ-MUSIC ver.3.0のワーク、ワークビュアZSV.Rについての解説を行います。

13．1．　演奏トラックワーク

[注意]

・文中の(.b)(.w)(.l)および(ds.b)(ds.w)(ds.l)などはデータ長を表している。

　(ds.b)/(.b)　バイトサイズ
　(ds.w)/(.w)　ワードサイズ(２バイト)
　(ds.l)/(.l)　ロングワードサイズ(４バイト)

・[ ]で表される数値はデフォルトであることを意味する。

・ひとつのワークアドレスに２つ以上のワークが設定されていることがある。

・具体的なワークへのアクセス法はリスト１、リスト２のようにして行う。

●リスト1

```
Z_MUSIC macro   func  *Z-MUSIC ver.3.0のファンクションコールマクロ
        moveq.l func,d0
        trap    #3
        endm

        moveq.l #n,d1               *トラック番号:0≦n≦65534
        Z_MUSIC #ZM_GET_PLAY_WORK

*アクセス例

        move.w  p_step_time(a0),d0  *トラックnのある瞬間の
                                    *ステップタイムを得る
```

●リスト2

```
Z_MUSIC macro   func  *Z-MUSIC ver.3.0のファンクションコールマクロ
        moveq.l func,d0
        trap    #3
        endm

        moveq.l #0,d1               *トラック番号＝1
        Z_MUSIC #ZM_GET_PLAY_WORK   *a0:トラック１の演奏トラッ
                                    *クワークの先頭アドレス
        move.l  a0,seq_wk_tbl       *ベースアドレスとなるので
                                    *これを格納

      *トラック12のワーク先頭アドレスを求めるとき
      movea.l seq_wk_tbl(pc),a0     *ベースアドレス取り出し
      moveq.l #11,d0                *トラック番号12
                                    *(内部表現では11)
      lsl.w   #trwk_size_,d0        *1トラック分の演奏トラック
                                    *ワークサイズ
      adda.w  d0,a0                 *a0:トラック12のワーク先頭
                                    *アドレス

*アクセス例
      move.w  p_step_time(a0),d0    *トラック12のある瞬間の
                                    *ステップタイムを得る

seq_wk_tbl:  ds.l   1               *ベースアドレス退避エリア
```

・ワークの不当な書き換えは、演奏に支障をきたすだけでなく、Z-MUSICの暴走を引き起こす可能性もあるため、細心の注意を必要とする。また、特殊な使用目的以外では、各ワークは読み出し専用と考え、書き換えはなるべく行わないようにしたほうがよい。

　各ワークは、必ずしも使用されているとは限らず、条件によっては、そのワークが持つ値はなんの意味も持たない場合がある。たとえばPMOD(ピッチモジュレーション,ビブラート)のスイッチp_pmod_sw(.b)がOFF(0)の場合、PMOD関連ワークに格納されている値はなんの意味も持たない。

　また、MMLやコマンドで与えたパラメータ値とワーク内部で表現される内部表現形式とは多少の食い違いがあるので注意すること。たとえば音色番号はMMLでは1〜32768であるが、ワーク(内部表現)では1を引いた0〜32767となっている。

・自作プログラムからZ-MUSIC ver.3.0のワークを参照する場合は以下に示すワークアドレス名 をラベル定義化した LABEL.MAC(ソースリストに同梱)をインクルードするとよい。

ノートワーク

```
p_note:         ds.b  4*16    *ノートオンテーブル
```

・ノートワークp_noteは以下の構成を取る

```
k_note          ds.b    1     *.b 発音したノートナンバー[-1]
                              *  (実際にKey OFFされるとd7=1)
k_velo:         ds.b    1     *.b 発音ベロシティ[-1]
                              *  (ゲートタイムが０になるとd7=1)
k_gate_time:    ds.w    1     *.w gate time[?]
                              *  (tieは1回の割り込み内では0,
                              *   割り込みを出ると-1)
```

基本パラメータ

```
p_step_time:    ds.w    1     *.w ステップタイム
p_track_stat:   ds.b    1     *.b トラックステータス
                              *  ($00:演奏中 $80:死亡 $01:終了
                              *   $02:停止中 $40:同期待ち)
p_track_mode:   ds.b    1     *.b トラックモード
                              *  ($80:Keyoffなし $00:通常)
p_trkfrq:       ds.b    1     *.b 割り込み頻度(マスタワーク)
p_trkfrq_wk:    ds.b    1     *.b 割り込み頻度(実行ワーク)
p_type:         ds.w    1     *.w 音源種別(FM:0 AD PCM:1
                              *          MIDI1〜4:$8000〜$8003
                              *          PATTERN:$7fff)
p_ch:           ds.w    1     *.w チャンネル番号(0〜32767)
p_data_pointer: ds.l    1     *.l 現在のコマンドポインタアドレス
p_now_pointer:  ds.l    1     *.l 現在実行中のZMDのアドレス
```

アフタータッチシーケンス関連

```
p_trk_seq_flg:  ds.b    1     *.b d7:VSEQ処理フラグ
p_aftc_n:       ds.b    1     *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽのﾎﾟｲﾝﾀ(-1,0〜7)
p_aftc_omt:     ds.b    1     *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ省略フラグ[0]
p_aftc_rltv:    ds.b    1     *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ相対値フラグ[0]
p_aftc_flg:     ds.b    1     *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ動作フラグ[0]
                              *d0 ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ 初めてか違うか
                              *  ([0]:初めて 1:違う)
                              *d5 ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ 休符時に同期する
                              *  か(1:Yes,[0]:No)
```

```
                                *d6 ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ ﾀｲ時に同期するか
                                *   (1:Yes,[0]:No)
                                *d7 ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ ｷｰｵﾝごとに
                                *同期するか([1]:Yes,0:No)
p_aftc_level:   ds.b    1       *.b 現在実行中のｱﾌﾀｰﾀｯﾁ値[-1]
p_aftc_tbl:     ds.b    8       *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ値テーブル*8
p_aftc_sw:      ds.b    1       *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽのｽｲｯﾁ
                                *   ([0]=OFF,$ff=1/8 MODE ON,
                                *    $01=OPTIONAL MODE ON)
p_aftc_sw2:     ds.b    1       *.b ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽのｽｲｯﾁ保存ワーク
                                *   ([0]=OFF,$ff=1/8 MODE ON,
                                *    $01=OPTIONALMODE ON)
p_aftc_1st_dly: ds.w    1       *.w ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽ初期ディレイ[0]
p_aftc_8st_tbl: ds.w    8       *.w ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽﾃﾞｨﾚｲ値*8
p_aftc_dly2:    ds.w    1       *.w ｱﾌﾀｰﾀｯﾁｼｰｹﾝｽﾃﾞｨﾚｲﾜｰｸ
p_aftc_syncnt:  ds.w    1       *.w 同期ｶｳﾝﾄ([1]～65535)
p_aftc_syncnt2: ds.w    1       *.w 同期ｶｳﾝﾄ実行ﾜｰｸ([1]～65535)
                ds.l    1
```

ピッチモジュレーション(ビブラート)関連

```
p_pmod_dpndpt:  ds.w    1       *.w PMOD振幅増減モードの加算量
                                * (-32768～[0]～32767)
p_pmod_dpt_tbl: ds.w    8       *8w PMOD振幅値ﾃｰﾌﾞﾙ [0]
p_pmod_dpnspd:  ds.w    1       *.w PMOD振幅増減モードの
                                *   振幅値加算周期([0]-32767)
p_pmod_spd_tbl: ds.w    8       *8w PMODｽﾋﾟｰﾄﾞ値ﾃｰﾌﾞﾙ[mod_dflt]
p_pmod_dpt_now: ds.w    1       *.w PMOD現在の実行振幅値[0]
p_pmod_wf:      ds.w    1       *.w PMOD波形タイプ[2]
                                *   プリセット波形:0～7
                                *   ユーザー波形:$8000+(0-32767)
p_pmod_1st_dly: ds.w    1       *.w PMOD初期ディレイ [0]
p_pmod_8st_tbl: ds.w    8       *8w PMODの振幅変化ディレイ
p_pmod_stp_tbl: ds.l    8       *8l PMOD加算ステップ値ﾃｰﾌﾞﾙ
                                *  (波形タイプ=矩形波,ノイズ,
                                *   波形ﾒﾓﾘでは振幅値が格納される)
p_pmod_sw:      ds.b    1       *.b PMOD･ｽｲｯﾁ
                                * ([0]=OFF,$ff=1/8 MODE ON,
                                * $01=OPTIONAL MODE ON)
p_pmod_sw2:     ds.b    1       *.b PMOD･ｽｲｯﾁ保存
                                *   ([0]=OFF,$ff=1/8 MODE ON,
                                *    $01=OPTIONAL MODE ON)
p_pmod_chain:   ds.b    1       *.b PMODの接続ﾌﾗｸﾞ
                                *   (-1:波形接続完了,0:即接続可能,
                                *     1:接続待機中,[2]:初回)
p_pmod_flg:     ds.b    1       *.b PMODの動作フラグ[0]
                                *d0 PMOD 初めてか違うか
                                *   ([0]:初めて,1:違う)
                                *d1 PMOD 振幅を増加減衰させるﾓｰﾄﾞ
                                *   ([0]:OFF,1:ON)
                                *d2 PMOD 波形接続可能か
                                *   ([0]:No,1:Yes)
                                *d3 PMOD 波形の値を休符のときも
                                *   更新([0]:No,1:Yes)
                                *d4 PMOD 波形の値をキーオンごとに
                                *   更新するモード([0]:OFF,1:ON)
                                *d5 PMOD 休符時に同期するか
                                *   (1:Yes,[0]:No)
                                *d6 PMOD タイ時に同期するか
                                *   (1:Yes,[0]:No)
                                *d7 PMOD ｷｰｵﾝごとに同期するか
                                *               ([1]:Yes,0:No)
p_pmod_mode:    ds.b    1       *.b MIDI PMODモード
                                *   ([-1]:通常,0:FM互換拡張,
                                *    1:MIDI互換拡張)
p_pmod_omt:     ds.b    1       *.b 1/8 MODE,OPTIONAL MODEの
                                *   省略ビットパターン
                                *   (振幅1個のときは[$01])
p_pitch_last:   ds.w    1       *.w 前回出力したベンダー値 [-1]
p_pmod_step2:   ds.l    1       *.l PMOD実行ｽﾃｯﾌﾟ(Fixed)
p_pmod_dly2:    ds.w    1       *.w PMOD･ﾃﾞｨﾚｲ実行ﾜｰｸ
p_pmod_spd:     ds.w    1       *.w PMOD･ｽﾋﾟｰﾄﾞ
p_pmod_spd2:    ds.w    1       *.w PMOD･ｽﾋﾟｰﾄﾞﾜｰｸ
p_pmod_pitch:   ds.l    1       *.l PMOD･ﾋﾟｯﾁﾜｰｸ(Fixed)
p_pmod_step_next: ds.l  1       *.l 次にくる予定のステップ
p_pmod_spd_next:  ds.w  1       *.w 次にくる予定のスピード
p_wvpm_start:   ds.l    1       *.l 波形メモリ開始アドレス
p_wvpm_point:   ds.l    1       *.l 波形メモリ現在のポインタ
p_wvpm_end:     ds.l    1       *.l 波形メモリ終了アドレス
                ds.b    1
p_wvpm_lpmd:    ds.b    1       *.b 波形メモリループモード
p_wvpm_lpst:    ds.l    1       *.l 波形メモリループ開始アドレス
p_wvpm_lped:    ds.l    1       *.l 波形メモリループ終了アドレス
p_wvpm_lptm:    ds.l    1       *.l 波形メモリループ回数(0=∞)
p_wvpm_lptm2:   ds.l    1       *.l 波形ﾒﾓﾘﾙｰﾌﾟ回数実行ﾜｰｸ(0=∞)
p_altp_flg:     ds.b    1       *.b 波形メモリ反復モードフラグ
p_pmod_n:       ds.b    1       *.b 8ﾎﾟｲﾝﾄ･PMOD･ﾃｰﾌﾞﾙﾎﾟｲﾝﾀ
                                *                     (-1,0～7)
p_pmod_wf2:     ds.w    1       *.w PMOD実行波形タイプ[2]
                                *   プリセット波形:0～7
                                *   ユーザー波形:$8000+(0-32767))
p_pmod_dpnrpt:  ds.w    1       *.w 振幅増減処理を何回行うか
                                * (0～32767,[0]:∞)
p_pmod_rndcnt:  ds.w    1       *.w 波形が何周したか(0-32767)
p_pmod_dpntime: ds.w    1       *.w 増減処理を何回行ったか
                                *                    (0-65535)
p_pmod_syncnt:  ds.w    1       *.w 同期ｶｳﾝﾄ([1]～65535)
p_pmod_syncnt2: ds.w    1       *.w 同期ｶｳﾝﾄ実行ﾜｰｸ([1]～65535)
                ds.l    16

p_arcc_param:   ds.b    __arcc_len*4    *ARCCワーク
                                        *(構成については後述)
p_vseq_param:   ds.b    __arcc_len      *ｴﾝﾊﾝｽﾄﾞﾍﾞﾛｼﾃｨｼｰｹﾝｽ･ﾜｰｸ
                                        *(構成については後述)
```

ポルタメント&ディチューン関係

```
p_detune:       ds.w    1       *.w デチューン
                                *(オートベンドの開始値にもなる)[0]
p_special_tie:  ds.w    1       *.w MIDI専用のスペシャルタイ用
                                *   ピッチワーク[0]
p_port_dly:     ds.w    1       *.w ポルタメントディレイ[0]
p_port_cnt:     ds.w    1       *.w ポルタメント継続時間[0]
p_bend_dly:     ds.w    1       *.w ベンドディレイ値[0]
p_bend_cnt:     ds.w    1       *.w ベンドテイル/ベンドタイム[0]
p_port2_dly:    ds.w    1       *.w ｵｰﾄﾎﾟﾙﾀﾒﾝﾄディレイ[0]
p_port2_cnt:    ds.w    1       *.w ｵｰﾄﾎﾟﾙﾀﾒﾝﾄﾃｲﾙ/ｵｰﾄﾎﾟﾙﾀﾒﾝﾄﾀｲﾑ[0]
p_port_pitch:   ds.l    1       *.l ﾎﾟﾙﾀﾒﾝﾄ/ｵｰﾄﾍﾞﾝﾄﾞ用
                                *   現在のベンド値*65536(Fixed)[0]
p_port_flg:     ds.b    1       *.b ﾎﾟﾙﾀﾒﾝﾄがｵﾝかｵﾌか
                                * ([0]=OFF,1=Normal Delay,
                                * -1=Minus Delay)
p_port2_flg:    ds.b    1       *.b ｵｰﾄﾎﾟﾙﾀﾒﾝﾄｽｲｯﾁ([0]=OFF,1=ON)
p_bend_sw:      ds.b    1       *.b ｵｰﾄﾍﾞﾝﾄﾞがｵﾝかｵﾌか
                                * ([0]=OFF,1=ON)
                                *   d7:初めてか(bit=0:初めて)
p_agogik_flg:   ds.b    1       *.b アゴーギクビットワーク[0]
                                *d0 Agogik主導権フラグ
                                *  ([0]=No/1=Yes)
                                *d2 Agogik 小節線発見時にリセット
                                *   するか (1:Yes,[0]:No)
                                *d3 Agogik 波形の値を休符のときも
                                *  更新([0]:No,1:Yes)
                                *d4 Agogik 波形の値をキーオンごと
                                *  に更新するモード([0]:OFF,1:ON)
                                *d5 Agogik 休符時に同期するか
                                *  (1:Yes,[0]:No)
                                *d6 Agogik タイ時に同期するか
                                *  (1:Yes,[0]:No)
                                *d7 Agogik ｷｰｵﾝごとに同期するか
                                *               (1:Yes,[0]:No)
p_bend_dst:     ds.w    1       *.w オートベンドのレンジ
                                *   (オートベンドの目的値となる)
                ds.w    1       *
```

```
p_port_step:    ds.l    1       *.l ﾎﾟﾙﾀﾒﾝﾄ用実行ステップ*65536
                ds.l    16
```

一般パラメータ

```
p_measure:      ds.l    1       *.l 現在の小節番号[0]
p_voice_rsv:    ds.w    1       *.w FM/AD PCM音源トラックのボイス
                                *リザーブ数
                                *  ([0]～7)
p_next_on:      ds.b    1       *.b FM/AD PCM音源トラックの次回の
                                *発音チャンネル [0]
p_timbre_set:                   *d7:Bank Selectの設定が行われたか
                                * (MIDI)
                                *d6:Programチェンジが行われたか
                                * (MIDI)
                                *d5:Volume設定が行われたか(MIDI)
                                *d4:Panpot設定が行われたか(MIDI)
                                *d3:Damper設定が行われたか(MIDI)
p_tone_set:     ds.b    1       *.b 第nﾋﾞｯﾄをチャンネルnとして
                                * 音色を設定したかどうかのフラグ
                                * (FM)[0]

p_return:       ds.l    1       *.l gosubの次のアドレス[0]

p_transpose:    ds.b    1       *.b キートランスポーズ[0]
p_mask_mode:    ds.b    1       *.b Mask Modeかどうか
                                *  ([0]=Normal,1=Masked)
p_damper:       ds.b    1       *.b damper on or off
                                *  ([0]～63:OFF/64～127:ON)
p_seq_flag:     ds.b    1       *.b []コマンド系の処理フラグビット
                                *パターン[0]
                                *d0:[D.C.]処理をしたことがあるか
                                * ([0]=No/1=Yes)
                                *d1:[FINE]処理をすべきかどうか
                                * ([0]=No/1=Yes)
                                *d2:[!]フラグ([0]=OFF/1=実行中)
p_bank_msb:     ds.b    1       *.b MIDI Bank MSB [-1]
p_bank_lsb:     ds.b    1       *.b MIDI Bank LSB [-1]
p_pgm:          ds.w    1       *.w Program Number([0]-32767)

p_sync_wk:      ds.b    1       *.b 強制同期コマンド用ワーク[1]
p_@b_range:     ds.b    1       *.b ベンドレンジ[12]
p_vol16:        ds.b    1       *.b 16段階音量(0-16),[-1]=無効
                                * (127段階音量モード)
p_velo16:       ds.b    1       *.b 16段階ベロシティ
                                * (0-16),[-1]=無効
                                *  (127段階ベロシティモード)
p_do_loop_flag: ds.b    1       *.b [DO]が以前に設定されているか
                                *   [0]=No/ループ回数(1～255)
p_md_flg:       ds.b    1       *.b 各種ビットワーク[0]
                                *d0 @B:ﾍﾞﾝﾄﾞ値をﾘｾｯﾄすべきかどうか
                                *   (MIDI専用 [0]=No/1=Yes)
                                *d1 @M:ﾓｼﾞｭﾚｰｼｮﾝ値をﾘｾｯﾄするかしな
                                *   いか(MIDI専用 [0]=No/1=Yes)
                                * ↑(それぞれ関係CTRLに0以外の値を
                                *   出力したときにビットが1になる)
                                *d2 AD PCM Variable Tuning Mode
                                *  ([0]:NOT)
                                *d3 OPMスロットマスクの切り換え
                                *d4 MIDI Tie Mode
                                *  ([0]:Normal,1:Special)
                                *d5 FM/AD PCMでVoice Reserveを
                                *行ったか([0]:No/1:Yes)
                                *d7 音色スプリットモード
                                *  ([0]:OFF,1:ON
p_how_many:     ds.b    1       *.b 今何音キーオンしているか
                                *   ([-1]:None/0～7:1～8)
p_last_note:    ds.b    1       *.b 前回発音したノート番号([-1])
```

MIDI専用関係

```
p_maker:        ds.b    1       *.b ﾒｰｶｰID(MIDI)[-1]
p_device:       ds.b    1       *.b ﾃﾞﾊﾞｲｽID(MIDI)[-1]
p_module:       ds.b    1       *.b ﾓｼﾞｭｰﾙID(MIDI)[-1]
p_effect1:      ds.b    1       *.b Effect Parameter 1(GS:RVB)[-1]
p_effect3:      ds.b    1       *.b Effect Parameter 3(GS:CHO)[-1]
p_effect4:      ds.b    1       *.b Effect Parameter 4(GS:DLY)[-1]
p_effect2:      ds.b    1       *.b Effect Parameter 2  [-1]
p_effect5:      ds.b    1       *.b Effect Parameter 5  [-1]
```

一般パラメータ

```
p_timbre_src:   ds.b    1       *.b 音色のコピー元チャンネルオフ
                                *   セット(FM:0-7)
p_sync:         ds.b    1       *.b OPM-LFOのシンクスイッチ
                                * (0=off,1=on)
p_om:           ds.b    1       *.b オペレータﾏｽｸ(&b0000-&b1111)
                ds.b    1

p_vol:          ds.b    1       *.b 127段階音量(0-127)  [??]
p_velo:         ds.b    1       *.b ベロシティ(0-[127])
p_frq:          ds.b    1       *.b AD PCMの周波数(0-6) [4]
p_pan:          ds.b    1       *.b Last PANPOT(0-127,-1=MUTE)[64]

p_onoff_bit:    ds.b    1       *キーオン/オフビットワーク
                                *d7:Key ONのときにbit=1が上書きさ
                                *   れる
                                *d6:Key OFFのときにbit=1が上書き
                                *   される
p_om_bak:       ds.b    1       *.b p_omのバックアップ
p_Q_gate:       ds.w    1       *.w Qコマンドの値(CONVERTER専用)
                                *   (0-256:NORMAL Q,負値:@Q)
p_midi_if:      ds.w    1       *.w ユーザーから見たMIDIインタフ
                                *ェイス番号(0-3)

p_do_loop_ptr:  ds.l    1       *.l [DO]のある位置
p_int_play_ope: ds.l    1       *.l 演奏処理ﾙｰﾁﾝｱﾄﾞﾚｽ
                                *通常はソースZMSC2.HAS中の
                                *int_play_ope_fm,
                                *int_play_ope_ad,int_play_ope_midi
p_midi_trans:   ds.l    1       *.l MIDI送信ルーチン

p_total:        ds.l    1       *.l トータルステップタイム
p_total_olp:    ds.l    1       *.l ﾙｰﾌﾟ外のﾄｰﾀﾙｽﾃｯﾌﾟﾀｲﾑ

p_split_tbl:    ds.b    48      *.l 音色スプリットの管理ワーク
                                *  音色スプリットの管理ワークは
                                *以下の構成を取る
                                *.b 割り当てる開始ノートナンバー
                                *   (0-127,128で残り全部)
                                *.b 割り当てる終端ノートナンバー
                                *   (0-127,128で残り全部)
                                *.w 割り当てるBANK番号
                                *   (MIDI:0-16383/FM,AD PCM:0-255)
                                *.w 割り当てる音色番号
                                *   (MIDI:0-127
                                *    FM,AD PCM:0-127 or 0-32767)
                                *以上6バイトが8つ続く
p_lfo_param:                    *以下2バイトペアワーク
p_pitch_param:  ds.w    1       *音程関係の特殊処理フラグ
p_level_param:  ds.w    1       *音量関係の特殊処理フラグ
p_key_on:       ds.l (16+1)*2   *これから発音する音の、
                                *ノート番号(.b),
                                *ベロシティ(.b),
                                *チャンネルオフセット(.w)
                                *音色番号(.l)
                                *が格納される。
                                *ただしチャンネルオフセットは
                                *FM/AD PCMのみで有効、
                                *MIDIでは常に0となる。
                                *たとえば p_ch の内容が3のとき
                                *チャンネルオフセットが2のとき、
                                *実際の発音はチャンネル5(=3+2)で
                                *行われる。p_ch の内容が 3のとき、
                                *チャンネルオフセットが6のとき、
                                *実際の発音はﾁｬﾝﾈﾙ2=(=(3+6) and 7)
```

```
                                *で行われる。
                ds.b    64
p_user:         ds.b    1       *.b ユーザー汎用ワーク
```

・ARCCとエンハンスドベロシティシーケンスは以下に示したARCCのワーク構成とまったく同じ構成を取る。なお、ARCC関連のワークは p_arcc_param から、エンハンスドベロシティシーケンスは p_vseq_param から開始される。

p_arcc_paramは、ARCCが1トラックにつき4つ実装されていることから、以下の構成のワークがARCC1からARCC4まで順番に4つ連続して並ぶことになる。

ARCC(AMOD)/ENHANCED VELOCITY SEQUENCE関連

```
p_arcc_wf:      ds.w    1       *.w ARCC波形タイプ
                                *  プリセット波形:0～7
                                *  ﾕｰｻﾞｰ波形:$8000+(0-32767))[2]
p_arcc_dpndpt:  ds.b    1       *.b ARCC振幅増減モードの加算量(-1
28～[0]～127)
p_arcc_dpt_tbl: ds.b    8       *.b ARCC振幅値ﾃｰﾌﾞﾙ*8 [0]
                ds.b    1
p_arcc_dpnspd:  ds.w    1       *.w ARECC振幅増減モードの
                                *   振幅値加算周期([0]-32767)
p_arcc_spd_tbl: ds.w    8       *.w ARCCｽﾋﾟｰﾄﾞ値ﾃｰﾌﾞﾙ[mod_dflt]
p_arcc_dpt_now: ds.b    1       *.b ARCC現在の実行振幅値[0]
p_arcc_flg2:    ds.b    1       *.b ARCC動作フラグ2
                                *d0 ARCC波形位相
                                *  ([0]:Normal,[1]:Reverse)
                                *d7 ARCC波形接続可能か
                                *  ([0]:No,1:Yes)
p_arcc_mode:    ds.b    1       *.b MIDI ARCCモード
                                *  ([-1]:通常,0～127:拡張)
p_arcc_omt:     ds.b    1       *.b 1/8 MODE,OPTIONAL MODEの省略
                                *ビットパターン[$01]
p_arcc_1st_dly: ds.w    1       *.w ARCC初期ディレイ [0]
p_arcc_8st_tbl: ds.w    8       *.w ARCCの振幅変化ディレイ
p_arcc_stp_tbl: ds.w    8       *.w ARCC加算ステップ値ﾃｰﾌﾞﾙ
                                * (波形タイプ=矩形波,ノイズ,波形
                                *  メモリでは振幅値が格納される)
p_arcc_step2:   ds.w    1       *.w ARCC実行ｽﾃｯﾌﾟ
p_arcc_spd:     ds.w    1       *.w ARCCｽﾋﾟｰﾄﾞ
p_arcc_spd2:    ds.w    1       *.w ARCCｽﾋﾟｰﾄﾞﾜｰｸ
p_arcc_dly2:    ds.w    1       *.w ARCCﾃﾞｨﾚｲﾜｰｸ
p_arcc_level:   ds.w    1       *.w ARCCﾎﾟｲﾝﾄﾜｰｸ
p_arcc_o:       ds.w    1       *.w ARCC波形スタート値
                                * (ノコギリ波,三角波専用)
p_arcc_sw:      ds.b    1       *.b ARCCスイッチ
                                *  ([0]=OFF,$ff=1/8 MODE ON,
                                *   $01=OPTIONAL MODE ON)
p_arcc_sw2:     ds.b    1       *.b ARCCスイッチ保存
                                *  ([0]=OFF,$ff=1/8 MODE ON,
                                *   $01=OPTIONAL MODE ON)
p_arcc_chain:   ds.b    1       *.b ARCCの接続ﾌﾗｸﾞ
                                *  (-1:波形接続完了,0:即接続可能,
                                *   1:接続待機中,[2]:初回))
p_arcc_flg:     ds.b    1       *.b ARCC動作フラグワーク
                                *d0 ARCC初めてか違うか
                                * ([0]:初めて,1:違う)
                                *d1 ARCC振幅を増加減衰させるﾓｰﾄﾞ
                                *  ([0]:OFF,1:ON)
                                *d2 ARCCのリセット要請
                                * ([0]:No [1]:Yes)
                                *d3 ARCC波形の値を休符の
                                *  ときも更新([0]:No,1:Yes)
                                *d4 ARCC波形の値をキーオンごとに
                                *  更新するモード([0]:OFF,1:ON)
                                *d5 ARCC 休符時に同期するか
                                *  (1:Yes,[0]:No)
                                *d6 ARCC タイ時に同期するか
                                *  (1:Yes,[0]:No)
                                *d7 ARCC ｷｰｵﾝごとに同期するか
                                *  ([1]:Yes,0:No)
                                *(ベロシティシーケンスのときのd6
                                *は以下のように扱われる。
                                *d6 VSEQ VELOCITYを与えられてる
                                *  ときも進行させるか
                                *  ([0]:no,1:yes)
p_arcc_last:    ds.b    1       *.b 前回出力したARCC値(MIDI)[-1]
p_arcc_reset:   ds.b    1       *.b ARCCのリセット値([0])
p_arcc_origin:  ds.b    1       *.b ARCC基準値([127])
p_arcc:         ds.b    1       *.b ARCCのコントロールナンバー
                                *  ([11]=Expression)
p_arcc_o_next:  ds.w    1       *.w 次にくる予定の波形開始点
p_arcc_step_next: ds.w  1       *.w 次にくる予定のステップ
p_arcc_spd_next:  ds.w  1       *.w 次にくる予定のスピード
p_wvam_start:   ds.l    1       *.l 波形メモリ開始アドレス
p_wvam_point:   ds.l    1       *.l 波形メモリ現在のポインタ
p_wvam_end:     ds.l    1       *.l 波形メモリ終了アドレス
                ds.b    1
p_wvam_lpmd:    ds.b    1       *.b 波形メモリループモード
p_wvam_lpst:    ds.l    1       *.l 波形メモリループ開始アドレス
p_wvam_lped:    ds.l    1       *.l 波形メモリループ終了アドレス
p_wvam_lptm:    ds.l    1       *.l 波形メモリループ回数(0=∞)
p_wvam_lptm2:   ds.l    1       *.l 波形ﾒﾓﾘﾙｰﾌﾟ回数実行ワーク
                                *   (0=∞)
p_alta_flg:     ds.b    1       *.b 波形メモリ反復モードフラグ
p_arcc_n:       ds.b    1       *.b ARCCﾃｰﾌﾞﾙﾎﾟｲﾝﾀ(-1,0～7)
p_arcc_wf2:     ds.w    1       *.w 次にくる予定の波形タイプ[2]
p_arcc_dpnrpt:  ds.w    1       *.w 振幅増減処理を何回行うか
                                *   (0～32767,[0]:∞)
p_arcc_rndcnt:  ds.w    1       *.w 波形が何周したか(0-32767)
p_arcc_dpntime: ds.w    1       *.w 増減処理を何回行ったか
                                *                  (0-65535)
p_arcc_syncnt:  ds.w    1       *.w 同期ｶｳﾝﾄ([1]～65535)
p_arcc_syncnt2: ds.w    1       *.w 同期ｶｳﾝﾄ実行ﾜｰｸ([1]～65535)
                ds.l    16
```

13．2．　バッファアドレステーブル

ファンクション ZM_GET_BUFFER_INFORMATION(MEASURE10参照)でZ-MUSIC ver.3.0の各バッファのアドレスを知ることができる。

ここでは、これらのバッファの構造や役割についての解説をする。

自作プログラムからZ-MUSIC ver.3.0のバッファアドレステーブルを参照する場合は以下に示すワーク名をラベル定義化した COMMON.MAC(ソースリストに同梱)をインクルードするとよい。

```
trk_buffer_top:         ds.l    1       *TRACKバッファ先頭ｱﾄﾞﾚｽ
trk_buffer_size:        ds.l    1       *TRACKバッファサイズ
trk_buffer_end:         ds.l    1       *TRACKバッファ最終ｱﾄﾞﾚｽ
adpcm_buffer_top:       ds.l    1       *(AD)PCMバッファ先頭ｱﾄﾞﾚｽ
adpcm_buffer_size:      ds.l    1       *(AD)PCMバッファサイズ
adpcm_buffer_end:       ds.l    1       *(AD)PCMバッファ最終ｱﾄﾞﾚｽ
wave_memory_top:        ds.l    1       *波形メモリﾊﾞｯﾌｧ先頭ｱﾄﾞﾚｽ
wave_memory_size:       ds.l    1       *波形メモリバッファサイズ
wave_memory_end:        ds.l    1       *波形メモリﾊﾞｯﾌｧ最終ｱﾄﾞﾚｽ
seq_wk_tbl:             ds.l    1       *音楽演奏ﾄﾗｯｸﾜｰｸ先頭ｱﾄﾞﾚｽ
seq_wk_tbl_se:          ds.l    1       *効果音用ﾄﾗｯｸﾜｰｸ先頭ｱﾄﾞﾚｽ
play_trk_tbl:           ds.l    1       *どのトラックを演奏するの
                                        *かを表すテーブル
play_trk_tbl_se:        ds.l    1       *どのトラックを演奏するの
                                        *かを表すテーブル(効果音)
adpcm_tbl:              ds.l    1       *(AD)PCMﾄｰﾝのｱﾄﾞﾚｽﾃｰﾌﾞﾙ[0]
adpcm_n_max:            ds.w    1       *定義できる最大トーン数[0]
adpcm_tbl2:             ds.l    1       *(AD)PCM音色の
                                        *        ｱﾄﾞﾚｽﾃｰﾌﾞﾙ[0]
adpcm_n_max2:           ds.w    1       *定義できる最大音色数[0]
wave_tbl:               ds.l    1       *波形ﾒﾓﾘﾃﾞｰﾀのﾃｰﾌﾞﾙｱﾄﾞﾚｽ
wave_n_max:             ds.w    1       *波形メモリの定義最大個数
fmsnd_buffer:           ds.l    1       *FM音色バッファ
fmsnd_n_max:            ds.w    1       *FM音色ﾊﾞｯﾌｧ定義最大個数
lyric_address:          ds.l    1       *歌詞文字列データアドレス
lyric_size:             ds.l    1       *歌詞文字列データサイズ
```

```
                                        *(文字列は0で終わらない
                                        *場合あり)
trk_po_tbl:             ds.l    1       *各トラックの先頭アドレス
                                        *格納テーブルアドレス
trk_po_tbl_se:          ds.l    1       *各トラックの先頭アドレス
                                        *格納テーブルアドレス
trk_n_max:              ds.w    1       *トラックの個数
pattern_trk:            ds.l    1       *ﾊﾟﾀｰﾝﾄﾗｯｸｱﾄﾞﾚｽ[0]
pattern_trk_se:         ds.l    1       *ﾊﾟﾀｰﾝﾄﾗｯｸｱﾄﾞﾚｽ効果音[0]
m_buffer_0:             ds.l    1       *CZ-6BM1-0用送信バッファ
m_buffer_1:             ds.l    1       *CZ-6BM1-1用送信バッファ
m_buffer_r:             ds.l    1       *RS-232C MIDI用の送信ﾊﾞｯﾌｧ
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
rec_buffer_0:           ds.l    1       *CZ-6BM1-0用受信バッファ
rec_buffer_1:           ds.l    1       *CZ-6BM1-1用受信バッファ
rec_buffer_r:           ds.l    1       *RS-232C-MIDI用の受信ﾊﾞｯﾌｧ
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
mm0_adr:                ds.l    1       *CZ-6BM1-0用のﾁｬﾝﾈﾙﾜｰｸ
mm1_adr:                ds.l    1       *CZ-6BM1-1用のﾁｬﾝﾈﾙﾜｰｸ
mmr0_adr:               ds.l    1       *RS-232C-MIDI A用ﾁｬﾝﾈﾙﾜｰｸ
mmr1_adr:               ds.l    1       *RS-232C-MIDI B用ﾁｬﾝﾈﾙﾜｰｸ
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
                        ds.l    1       *RESERVED
n_of_err:               ds.l    1       *発生しているエラーの数
err_stock_next:         ds.l    1       *次のエラー格納アドレス
err_stock_addr:         ds.l    1       *ｴﾗｰｽﾄｯｸﾊﾞｯﾌｧ先頭アドレス
err_stock_size:         ds.l    1       *ｴﾗｰｽﾄｯｸバッファサイズ
```

●トラックバッファ

トラックバッファはZMDを格納するために確保されている。

外部アプリケーションで確保したメモリ領域をZ-MUSIC ver.3.0から参照させるには、

(trk_buffer_top)＝メモリ領域の先頭アドレス
(trk_buffer_size)＝メモリ領域のサイズ
(trk_buffer_end)＝メモリ領域の最終アドレス

を設定する。

●演奏トラックワーク

演奏中の演奏データの各トラックのワーク。Z-MUSIC ver.3.0はver.2.0とは異なり演奏に使用されているトラック数しか演奏トラックワークは存在しない。１本のトラックにつき、Z-MUSIC ver.3.0のソースリストLABEL.MAC中に定義されているtrwk_sizeの数だけ(ver.3.0時点で2048)バイトのワークが割り当てられている。各ワークの役割と構成については13.1.を参照のこと。

(seq_wk_tbl)＋トラック番号(0〜)×trwk_size

で任意のトラックのワーク先頭アドレスを求めることができる。

●効果音トラックワーク

効果音の演奏トラックワーク。トラック数はZMSC3.X組み込み時の'-F'スイッチで指定された数だけ存在する。役割と構成、ワークサイズはまったく演奏トラックワークと同等。

●(AD)PCMバッファ

(AD)PCMバッファにはZPDなどの(AD)PCMデータが格納される。

外部アプリケーションで確保したメモリ領域をZ-MUSIC ver.3.0から参照させるには、

(adpcm_buffer_top)＝メモリ領域の先頭アドレス
(adpcm_buffer_size)＝メモリ領域のサイズ
(adpcm_buffer_end)＝メモリ領域の最終アドレス

を設定する。

●(AD)PCM管理テーブル

adpcm_tblは音色番号1〜256,ノート番号0〜127に割り当てられた(AD)PCMトーンを管理するワーク。

adpcm_tbl2はバンク0〜255,音色番号1〜128に割り当てられた(AD)PCM音色を管理するワーク。

音色番号1のノート番号0からノート番号127まで順番に以下の構成が繰り返す。音色番号1のノート番号127の次は音色番号2のノート番号0からノート番号127までが続く。以下音色番号256までこれが繰り返されている。

ただし、実際に(AD)PCMデータが格納されているノート番号を含むバンクまでしか領域は保証されない。

(AD)PCM音色についても、バンク0の音色番号1〜128からバンク255の音色番号1〜128までが同様の構成を取る。

```
adt_type:               ds.b    1       *データタイプ(0=empty,
                                        *   -1=AD PCM,1=16bitPCM)
adt_orig_key:           ds.b    1       *オリジナルキー:0-127
                        ds.b    1       *Reserved
                        ds.b    1       *Reserved
adt_addr:               ds.l    1       *データアドレス
adt_size:               ds.l    1       *サイズ
adt_loop_start:         ds.l    1       *ループ開始ポイント
adt_loop_end:           ds.l    1       *ループ終端ポイント
adt_loop_time:          ds.l    1       *ループ回数
                        ds.b    8       *Reserved
adt_name:               ds.b    32      *コメント
```

ひとつの(AD)PCMデータに対して割り当てられるワークサイズが64バイトであることから任意の音色番号b(1〜256)、ノート番号n(0〜127)に対応する(AD)PCMトーンの管理ワークアドレスは、

(adpcm_tbl)＋(b−1)×128×64＋n×64

で求めることができる。同様に任意のバンク番号b(0〜255)、音色番号tの(AD)PCM音色の管理ワークアドレスは、

(adpcm_tbl2)＋b×128×64＋t×64

で求めることができる。

自作プログラムから上記のオフセットテーブルを利用して(AD)PCM管理テーブルを参照する場合はソースリストに含まれる TABLE.MAC(ソースリストに同梱) をインクルードするとよい。

●波形メモリ管理テーブル

wave_tblは波形番号8〜32767に割り当てられた波形メモリデータ列を管理するワーク。1つの波形につき、以下のようなワーク構成を取っている。これが、連続して波形番号８から波形番号32767まで続く。ただし、wave_n_maxの値で表される音色番号の分までしかバッファは確保されていない。

```
wave_start_addr:        ds.l    1       *波形の開始アドレス(.l)
wave_end_addr:          ds.l    1       *波形の最終アドレス+2(.l)
                        ds.b    1       *Reserved
wave_loop_type:         ds.b    1       *ループタイプ(.w)
wave_loop_start:        ds.l    1       *ループ開始アドレス(.l)
wave_loop_end:          ds.l    1       *ループ最終アドレス+2(.l)
wave_loop_time:         ds.l    1       *ループ回数
                        ds.l    1       *Reserved
                        ds.b    1       *Reserved
wave_name_len:          ds.b    1       *波形名称文字列長(.w)
wave_name_addr:         ds.l    1       *波形名称文字列格納ｱﾄﾞﾚｽ(.l)
```

なお、波形のアドレスなどはすべて絶対番地で表されている。ループタイプ値は0〜2で与えたものが −1,0,+1 にシフトされている点に注意。

ひとつの波形に対して割り当てられるワークサイズは64バイトであることから任意の波形番号w(8〜32767)に対応したワークアドレスは、

(wave_tbl)＋(w-8)×64

にて求めることができる。

自作プログラムから上記のオフセットテーブルを利用して波形メモリ管理テーブルを参照する場合はソースリストに含まれる TABLE.MAC(ソースリストに同梱) をインクルードするとよい。

●FM音源音色バッファ

fmsnd_bufferはFM音源音色を格納するバッファで、音色データが以下に示すようなフォーマットで格納されている。バンク0の音色番号1からバン

ク255の音色番号128(または音色番号1から32768)までが順番に続く。ただし、fmsnd_n_maxの値で表される音色番号の分までしかバッファは確保されていない。

```
fm_flag:          ds.w    1       *音色有効フラグ
fm_LFRQ:          ds.b    1       *LFRQ
fm_PMD:           ds.b    1       *PMD
fm_AMD:           ds.b    1       *AMD
fm_SYNC_OM_WF:    ds.b    1       *d0-d1:WF,d3-d6:OM,d7:SYNC
fm_PAN_AF:        ds.b    1       *d0-d5:AF,d6-d7:PAN
fm_PMS_AMS:       ds.b    1       *d0-d1:AMS,d4-d6:PMS
fm_OP1_DT1_MUL:   ds.b    1       *OP1 d0-d3:MUL,d4-d7:DT1
fm_OP3_DT1_MUL:   ds.b    1       *OP3 d0-d3:MUL,d4-d7:DT1
fm_OP2_DT1_MUL:   ds.b    1       *OP2 d0-d3:MUL,d4-d7:DT1
fm_OP4_DT1_MUL:   ds.b    1       *OP4 d0-d3:MUL,d4-d7:DT1
fm_OP1_TL:        ds.b    1       *OP1 TL
fm_OP3_TL:        ds.b    1       *OP3 TL
fm_OP2_TL:        ds.b    1       *OP2 TL
fm_OP4_TL:        ds.b    1       *OP4 TL
fm_OP1_KS_AR:     ds.b    1       *OP1 d0-d4:AR,d6-d7:KS
fm_OP3_KS_AR:     ds.b    1       *OP3 d0-d4:AR,d6-d7:KS
fm_OP2_KS_AR:     ds.b    1       *OP2 d0-d4:AR,d6-d7:KS
fm_OP4_KS_AR:     ds.b    1       *OP4 d0-d4:AR,d6-d7:KS
fm_OP1_AME_1DR:   ds.b    1       *OP1 d0-d4:D1R,d7:AMS-EN
fm_OP3_AME_1DR:   ds.b    1       *OP3 d0-d4:D1R,d7:AMS-EN
fm_OP2_AME_1DR:   ds.b    1       *OP2 d0-d4:D1R,d7:AMS-EN
fm_OP4_AME_1DR:   ds.b    1       *OP4 d0-d4:D1R,d7:AMS-EN
fm_OP1_DT2_2DR:   ds.b    1       *OP1 d0-d4:D2R,d6-d7:DT2
fm_OP3_DT2_2DR:   ds.b    1       *OP3 d0-d4:D2R,d6-d7:DT2
fm_OP2_DT2_2DR:   ds.b    1       *OP2 d0-d4:D2R,d6-d7:DT2
fm_OP4_DT2_2DR:   ds.b    1       *OP4 d0-d4:D2R,d6-d7:DT2
fm_OP1_D1L_RR:    ds.b    1       *OP1 d0-d3:RR,d4-d7:D1L
fm_OP3_D1L_RR:    ds.b    1       *OP3 d0-d3:RR,d4-d7:D1L
fm_OP2_D1L_RR:    ds.b    1       *OP2 d0-d3:RR,d4-d7:D1L
fm_OP4_D1L_RR:    ds.b    1       *OP4 d0-d3:RR,d4-d7:D1L
fm_NAME:          ds.b    1       *音色名文字列(最大16文字)
```

1音色につき48バイトをとるので任意のバンク番号b(0-255)、音色番号t(1-128)の音色データのアドレスは、

　(fmsnd_buffer)+b×128×48+(t-1)*48

で求められる(バンク番号を用いないで指定する場合は、上の式中の「b×128×48」を省いて考える)。

自作プログラムから上記のオフセットテーブルを利用してFM音源音色バッファを参照する場合はソースリストに含まれる TABLE.MAC(ソースリストに同梱) をインクルードするとよい。

●各トラックの先頭アドレス

各トラックの先頭アドレスはtrk_po_tblに格納されているアドレスから、

トラック１の先頭アドレス(.L)
トラック２の先頭アドレス(.L)
トラック３の先頭アドレス(.L)
　　：
　　：

のように格納されている。(効果音トラックの場合はtrk_po_tbl_se)

よって、任意のトラック番号t(0～65534)のトラックの先頭アドレスは、

　(trk_po_tbl)+t×4

で求めることができる。

●演奏に使用されるトラック

実際の演奏に使用されているトラック番号(0～65534)は、play_trk_tblに格納されているアドレスから、$ffff(.w)を終端コードとし、ワードサイズで列記されている。たとえばトラック番号1～8が演奏されているとするとplay_trk_tblに格納されているアドレスから、

　$0000 $0001 $0002 $0003 $0004 $0005 $0006 $0007 $ffff

のような値が格納されている。トラック番号1～8が0～7になっている点に注意。

１３．３．　Z-MUSICの状態ワーク

ファンクション ZM_GET_ZMSC_STATUS(MEASURE10参照)でZ-MUSIC ver.3.0の状態ワーク(ステータスワーク)のアドレスを知ることができる。状態ワークはその時点におけるZ-MUSICの状態を表したワークエリアで、フェーダー関連のパラメータやカレントMIDIインタフェイス、テンポなどのグローバルパラメータなどもここに配置されている。

ここでは、これらのワークの構造や役割についての解説をする。

自作プログラムからZ-MUSIC ver.3.0の状態ワークを参照する場合は以下に示す状態ワーク名をラベル定義化した ZM_STAT.MAC(ソースリストに同梱) をインクルードするとよい。

```
midi_board:             dc.b    0         *MIDI I/F有効フラグ
                                          *d0～d3がそれぞれ
                                          *CZ6BM1#1,#2,
                                          *RS232C-MIDI A,Bに対応
available_device:       dc.b    0         *実際に使用するﾃﾞﾊﾞｲｽﾌﾗｸﾞ
                                          *(midi_boardと同書式)
timer_flg:              dc.b    0         *TIMER ON/OFF
                                          *(d0:TIMER A,
                                          *d1:TIMER B,
                                          *d2:YM3802 TIMER)
                                          *(d7はFUNC INTERCEPT_PLAY
                                          *が有効時にm_playが実行さ
                                          *れると1に上書きされる。
                                          *外部ｱﾌﾟﾘ向け専用ﾜｰｸﾋﾞｯﾄ)
                        dc.b    0
wk_size:dc.w            trwk_size         *1トラックワークのサイズ
play_start_time:        dc.l    -1        *演奏開始時刻(10時23分52秒
                                          *なら$00102352,初期値=-1)
play_stop_time:         dc.l    -1        *演奏停止時刻 (10時23分5秒
                                          *なら$00102352,初期値=-1)
play_cont_time:         dc.l    -1        *演奏再開時刻(10時23分52秒
                                          *なら$00102352,初期値=-1)
dev_end_adr:            ds.l    1         *ZMSC3.X最終アドレス

t_min:                  dc.w    10        *音楽演奏用テンポの最小値
t_max:                  dc.w    32767     *音楽演奏用テンポの最大値
t_min_se:               dc.w    20        *効果音演奏用テンポ最小値
t_max_se:               dc.w    5000      *効果音演奏用テンポ最大値
_@t_max:                dc.w    $3fff     *音楽演奏用タイマ値最大値
_@t_max_se:             dc.w    $ff       *効果音演奏用ﾀｲﾏ値最大値
mask_opm_ch:            dc.w    0         *FM音源のﾏｽｸﾁｬﾝﾈﾙ対応ﾋﾞｯﾄ
                                          *d0～d7がチャンネル1～8に
                                          *対応。対応ビット=1:Masked
se_tr_max:              dc.w    0         *効果音の最大トラック数[0]
ext_pcmdrv:             dc.b    0         *外部PCMドライバ
                                          *([0]:なし
                                          *   1:組み込まれている)
current_rs232c_part:    dc.b    1         *RS-232Cの現在の選択パート
                                          *0:PART A,$FF:PART B,
                                          *[1]:未定義
current_midi_in_r:      dc.w    -1        *カレント相対I/F番号
                                          * (USERから見た0-3)
current_midi_in_w:      dc.b    -1        *通常は常時0
current_midi_in_b:      dc.b    -1        *カレント絶対I/F番号
                                          * (内部で使用する(0-3)*2)
current_midi_out_r:     dc.w    -1        *カレント相対I/F番号
                                          * (USERから見た0-3)
current_midi_out_w:     dc.b    -1        *通常は常時0
current_midi_out_b:     dc.b    -1        *カレント絶対I/F番号
                                          * (内部で使用する(0-3)*2)
midi_if_tbl:            dcb.b   8+1,-1    *MIDIｲﾝﾀﾌｪｲｽの使用設定
                                          *(0:CZ-6BM1-1,2:CZ-6BM1-2,
                                          *4:RS-232C-A,
                                          *6:RS-232C-B,-1:なし)
ch_wk_mode:             dc.b    -1        *ﾁｬﾝﾈﾙﾜｰｸの使用状態
                                          *-1=全部(DEFAULT),
                                          * 0=FMなし(通常禁止)
```

```
                                        *1=MIDIなし,
                                        *2=全部なし(通常禁止)
gs_id:          dc.b    $10,$10,$10,$10 *DEVICE IDデフォルト値
                ds.b    4               *Reserved
sc88_id:        dc.b    $10,$10,$10,$10
                ds.b    4               *Reserved
mt32_id:        dc.b    $10,$10,$10,$10 *
                ds.b    4               *Reserved
u220_id:        dc.b    $10,$10,$10,$10 *
                ds.b    4               *Reserved
m1_id:          dc.b    $30,$30,$30,$30 *
                ds.b    4               *Reserved
adpcm_frq:      dc.b    0               *AD PCM周波数
adpcm_pan:      dc.b    0               *AD PCMパンポット
adpb_clr:       dc.b    0               *AD PCMﾊﾞｯﾌｧ初期化有無ﾌﾗｸﾞ
                                        *(0以外=要初期化)
wvmm_clr:       dc.b    0               *波形ﾒﾓﾘﾊﾞｯﾌｧ初期化有無ﾌﾗｸﾞ
                                        *(0以外=要初期化)

meter:          dc.w    $0404           *拍子
metronome:      dc.b    0               *Reserved
                dc.b    0               *Reserved
key:            dc.w    0               *調号
mst_clk:        dc.w    192             *全音符の絶対音長カウント
tempo_value:    dc.w    120             *テンポ値(DEFAULT=120)
timer_value:    dc.w    0               *タイマの値

meter_se:       dc.w    $0404           *拍子(効果音サイド)
metronome_se:   dc.b    0               *Reserved
                dc.b    0               *Reserved
key_se:         dc.w    0               *調号
mst_clk_se:     dc.w    192             *全音符の絶対音長カウント
                                        * (効果音サイド)
tempo_value_se  dc.w    120             *テンポ値(DEFAULT=120)
timer_value_se: dc.w    0               *タイマの値
eox_w:          dcb.w   8,3             *EOXウェイト(DEFAULT=3)
se_mode:        dc.b    0               *AD PCMの効果音モードか
                                        * (0=No,0以外=Yes)
fader_flag:     dc.b    0               *現在実行中のフェーダー
                                        *(d7:マスターフェーダー,
                                        * d6:チャンネルフェーダー)
                                        *実行中のフェーダーに対応
                                        *したビット=1でそのフェー
                                        *ダーが可動中
fnc_no:         dc.w    $7fff           *最後に実行した(あるいは
                                        *現在実行中の)ﾌｧﾝｸｼｮﾝ番号
sr_type:        dc.w    $2500           *SRをﾏｽｸする際のﾃﾞﾌｫﾙﾄ値
                                        *(普通は書き変わる)
sp_buf:         dc.l    0               *ファンクション実行時の
                                        *スタック保存ワーク
fnc_quit_addr:  dc.l    0               *緊急帰還アドレス(エラーが
                                        *発生した場合の脱出ｱﾄﾞﾚｽ)
opmset_bsr_ms:  dc.l    opmset          *FM音源ｱｸｾｽ ﾙｰﾁﾝ ｱﾄﾞﾚｽ
                                        * (効果音モード時opmset_se)
timer_mode:     dc.b    0               *どのタイマを使うか
                                        *タイマA:-1,タイマB:0,
                                        *YM3802タイマ:1
synchro_mode:   dc.b    0               *外部同期モードか
                                        * (0=No,$FF=Yes)
polypress_mode: dc.b    -1              *ﾎﾟﾘﾌｫﾆｯｸﾌﾟﾚｯｼｬｰﾓｰﾄﾞ
                                        * ([-1]:ON,0=No)
no_init_mode:   dc.b    0               *初期化なしモードか
                                        * ([0]=No,0以外=Yes)
zmusic_int:     dc.l    0               *Z-MUSICが割り込むと
                                        *インクリメントされる
last_zmusic_int:dc.l    0               *前回MIDI_INP1を実行した
                                        *ときのzmusic_intの値
zpd_last_fn:    ds.b    54              *前回読み込んだﾌｧｲﾙの名前
header_buffer:  ds.b    8               *ファイルID用バッファ
fm_tune_tbl:    dc.l    0               *FM音源周波数調整テーブル
```

```
pcm_tune_tbl:   dc.l    0               *PCM音源周波数調整ﾃｰﾌﾞﾙ
occupy_flag:    dc.b    0               *占有状態
                                        *(0=FREE,1=占有されている)
cmd_or_dev:     dc.b    0               *-1=コマンドライン,
                                        * 0=DEVICE
errmes_lang:    dc.b    0               *メッセージ表示言語
                                        * ([0]:English,1:Japanese)
external_applications:  dc.b    0       *外部アプリの常駐状況
                                        *d7:ZP -d,d6:ZP -j
                                        *対応ビット=1のとき
                                        *そのｱﾌﾟﾘｹｰｼｮﾝが
                                        *組み込まれている
smf_end_flag:   dc.b    0               *SMF 転送終了フラグ
rs232c_mode:    dc.b    0               *RS232C-MIDIのモード
                                        *d7=(0:通常,1:デュアル)
                                        *d0-d3:ボーレート時定数[3]
perform_flg:    dc.b    0               *ZM_PLAY,ZM_CONT,ZM_STOP
                                        *のﾌｧﾝｸｼｮﾝが実行されるた
                                        *びに対応ﾋﾞｯﾄが1に上書き
                                        *される
                                        *d0:PLAY d1:CONT d2:STOP
                                        *d7:loop mode(0:normal
                                        * 1:no loop/FUNC $59参照)
v2_compatch:    dc.b    0               *V2ｺﾝﾊﾟﾁﾓｰﾄﾞか
                                        * [0]:No 1:Yes
                .even
fm_vol_tbl:     ds.b    16              *ﾌｪｰﾄﾞｱｳﾄ/ｲﾝ用の音量ﾜｰｸ
ad_vol_tbl:     ds.b    16
m0_vol_tbl:     ds.b    16
m1_vol_tbl:     ds.b    16
mr0_vol_tbl:    ds.b    16
mr1_vol_tbl:    ds.b    16

*以下spd,spd2,lvlw,lvlb系のワークの配置はソースリストZMSC2.HASの都
合に合わせてある
                                        *ﾏｽﾀｰﾌｪｰﾀﾞｰﾊﾟﾗﾒｰﾀ群
mstfd_fm_spd2:  ds.w    1               *スピード実行ワーク(.w)
mstfd_fm_spd:   ds.w    1               *スピード値(.w)
mstfd_fm_lvlw:  ds.b    1               *フェーダーレベル(.w)
mstfd_fm_lvlb:  ds.b    1               *フェーダーレベル(.b)
mstfd_fm_dest:  ds.b    1               *目的値(.b)
mstfd_fm_mode:  ds.b    1               *フェーダーモード(.b)

mstfd_ad_spd2:  ds.w    1               *スピード実行ワーク(.w)
mstfd_ad_spd:   ds.w    1               *スピード値(.w)
mstfd_ad_lvlw:  ds.b    1               *フェーダーレベル(.w)
mstfd_ad_lvlb:  ds.b    1               *フェーダーレベル(.b)
mstfd_ad_dest:  ds.b    1               *目的値(.b)
mstfd_ad_mode:  ds.b    1               *フェーダーモード(.b)

mstfd_m0_spd2:  ds.w    1               *スピード実行ワーク(.w)
mstfd_m0_spd:   ds.w    1               *スピード値(.w)
mstfd_m0_lvlw:  ds.b    1               *フェーダーレベル(.w)
mstfd_m0_lvlb:  ds.b    1               *フェーダーレベル(.b)
mstfd_m0_dest:  ds.b    1               *目的値(.b)
mstfd_m0_mode:  ds.b    1               *フェーダーモード(.b)

mstfd_m1_spd2:  ds.w    1               *スピード実行ワーク(.w)
mstfd_m1_spd:   ds.w    1               *スピード値(.w)
mstfd_m1_lvlw:  ds.b    1               *フェーダーレベル(.w)
mstfd_m1_lvlb:  ds.b    1               *フェーダーレベル(.b)
mstfd_m1_dest:  ds.b    1               *目的値(.b)
mstfd_m1_mode:  ds.b    1               *フェーダーモード(.b)

mstfd_mr0_spd2: ds.w    1               *スピード実行ワーク(.w)
mstfd_mr0_spd:  ds.w    1               *スピード値(.w)
mstfd_mr0_lvlw: ds.b    1               *フェーダーレベル(.w)
mstfd_mr0_lvlb: ds.b    1               *フェーダーレベル(.b)
mstfd_mr0_dest: ds.b    1               *目的値(.b)
mstfd_mr0_mode: ds.b    1               *フェーダーモード(.b)
```

```
mstfd_mr1_spd2: ds.w    1           *スピード実行ワーク(.w)
mstfd_mr1_spd:  ds.w    1           *スピード値(.w)
mstfd_mr1_lvlw: ds.b    1           *フェーダーレベル(.w)
mstfd_mr1_lvlb: ds.b    1           *フェーダーレベル(.b)
mstfd_mr1_dest: ds.b    1           *目的値(.b)
mstfd_mr1_mode: ds.b    1           *フェーダーモード(.b)

ch_fm_fdp:      ds.b    fd_wkl*16   *ﾁｬﾝﾈﾙﾌｪｰﾀﾞｰﾊﾟﾗﾒｰﾀ群
ch_ad_fdp:      ds.b    fd_wkl*16   *(1ch:8byte構成*16ch分)
ch_m0_fdp:      ds.b    fd_wkl*16
ch_m1_fdp:      ds.b    fd_wkl*16
ch_mr0_fdp:     ds.b    fd_wkl*16
ch_mr1_fdp:     ds.b    fd_wkl*16

ch_fader_tbl:   ds.w    96      *チャンネルフェーダー
                                *(フェード処理するチャンネル番号
                                * 0-95が並ぶ)
                dc.w    -1      *End code
master_fader_tbl:               *マスターフェーダー
                                *(フェード処理するデバイス番号*fd
_wklが並ぶ)
                ds.w    6       *0,8,16,24,32,40
                dc.w    -1      *End code

ch_mask_fm:     dc.w    0       *ﾁｬﾝﾈﾙﾌｪｰﾀﾞｰﾋﾞｯﾄﾜｰｸ
ch_mask_ad:     dc.w    0
ch_mask_m0:     dc.w    0
ch_mask_m1:     dc.w    0
ch_mask_mr0:    dc.w    0
ch_mask_mr1:    dc.w    0
                .even
agogik_work:    ds.b    __pmod_len *音楽演奏用ｱｺﾞｰｷﾞｸﾜｰｸ
agogik_work_se: ds.b    __pmod_len *効果音演奏用ｱｺﾞｰｷﾞｸﾜｰｸ
                .even
mpcm_vol_tbl:           *MPCM.X用音量テーブル64=原音レベル
                dc.w      0,   3,   6,   9,  12,  15,  18,  21
                dc.w     24,  25,  26,  27,  28,  29,  30,  31
                dc.w     32,  33,  34,  35,  36,  37,  38,  39
                dc.w     40,  41,  42,  43,  44,  45,  46,  47
                dc.w     48,  50,  52,  54,  56,  58,  60,  62
                dc.w     64,  66,  68,  70,  72,  74,  76,  78
                dc.w     80,  82,  84,  86,  88,  90,  92,  94
                dc.w     96, 100, 104, 108, 112, 116, 120, 124
                dc.w    128, 132, 136, 140, 144, 148, 152, 156
                dc.w    160, 164, 168, 172, 176, 180, 184, 188
                dc.w    192, 200, 208, 216, 224, 232, 240, 248
                dc.w    256, 264, 272, 280, 288, 296, 304, 312
                dc.w    320, 328, 336, 344, 352, 360, 368, 376
                dc.w    384, 400, 416, 432, 448, 464, 480, 496
                dc.w    512, 528, 544, 560, 576, 592, 608, 624
                dc.w    640, 656, 672, 688, 704, 720, 736, 752
_result_start:                          *戻り値バッファ
mask_track_result:      ds.l    1       *4*(trk_n_max+1) bytes
mask_channel_result     ds.l    1       *4*(trk_n_max+1) bytes
m_play_result:          ds.l    1       *4*(trk_n_max+1) bytes
m_stop_result:          ds.l    1       *4*(trk_n_max+1) bytes
m_cont_result:          ds.l    1       *4*(trk_n_max+1) bytes
                                        *SMF再生関連ワーク
smf_delta:      ds.l    1               *SMF デルタタイム
smf_running:    ds.l    1               *SMFﾗﾝﾆﾝｸﾞｽﾃｰﾀｽｶｳﾝﾀ
smf_pointer:    ds.l    1               *SMF データポインタ
smf_transfer:   ds.l    1               *SMF MIDI送信エントリ
smf_mst_clk:    ds.l    1               *SMFマスタークロック
smf_addr:       ds.l    1               *SMFの格納アドレス

fo_ch_fm:       ds.b    8       *フェードアウト/インのﾜｰｸ
                ds.b    8       *DUMMY
fo_ch_ad:       ds.b    16      *d0:ch fader active flag 01
fo_ch_m0:       ds.b    16      *d1:master fader active flag 02
fo_ch_m1:       ds.b    16      *d5:master fader move flag 20
fo_ch_mr0:      ds.b    16      *d6:ch fader move flag 40
fo_ch_mr1:      ds.b    16      *d7:処理したかどうか 80
work:
pt_ptr:         ds.l    1       *play_trk_tblの指標
                                *必ずwork=pt_ptr
cf              ds.b    8       *ｵﾍﾟﾚｰﾀのｺﾈｸｼｮﾝﾃｰﾌﾞﾙﾋﾞｯﾄ
ol1:            ds.b    8       *アウトプットレベル1
ol2:            ds.b    8       *アウトプットレベル2
ol3:            ds.b    8       *アウトプットレベル3
ol4:            ds.b    8       *アウトプットレベル4
opm_kon:        ds.b    8       *OPMレジスタ8番のワーク
opm_nom:        ds.b    8       *オペレータマスク(OM)のNOT
                                *(&b11110000-&b11111111)
_opm:           ds.b    256     *OPMのﾚｼﾞｽﾀｲﾒｰｼﾞﾜｰｸ
opm_pmd:dc.b    0
```

Z-MUSIC状態ワークの不当な書き替えは、演奏に支障をきたすだけでなく、Z-MUSICの暴走を引き起こす可能性もあるため、細心の注意を必要とする。また、特殊な使用目的以外では、各ワークは読み出し専用と考え、書き替えはなるべく行わないようにしたほうがよい。

また、ワーク構成はバージョンによっては変更が施される可能性もあるため、自作プログラムからZ-MUSIC状態ワークを参照する場合にはZ-MUSICのバージョンのチェックを厳しく行ったほうがよい。

13．4．ワークビュアZSV.R

ZSV.RはZ-MUSIC ver3.0以上で動作するワーク/ステータスビュアである(ZSV.RはZ-MUSIC STATUS VIEWRの略)。

ZMSやZMDなどの演奏データを演奏したあと、

A>ZSV

として起動すると、演奏中に変化するZ-MUSIC内部のパラメータの値をリアルタイムに見ることができる。

13．4．1．ZSV.Rの表示パラメータ

13．4．1．1．コモンパラメータ

画面一番上部には現在演奏中の曲のタイトル文字列が表示され、その下にその曲全体の動作を表すパラメータ群が表示されている。このパラメータ群を特にコモンパラメータとよぶ。

以下にコモンパラメータの一覧を示す。

METER:	拍子
KEY:	調号
TEMPO:	テンポ
PLAYTIME:	演奏開始からの経過時間/演奏時間
右上部のゲージ	演奏の進行度を表している。右が終端。 終結する曲は曲の終わりが、 リピート曲は1ループが、ゲージ終端に相当する。
AGOGIK:	アゴーギク(テンポの緩急)の振幅
WAVE:	アゴーギクの波形種別
VARIANT:	アゴーギク振幅の変化量
REPEAT:	現時点での演奏反復回数
FMOPM:	現在演奏中の曲のFM音源使用チャンネル数
FMOPM FADER:	FM音源部のマスターフェーダー
AD PCM:	現在演奏中の曲のAD PCM音源使用チャンネル数
AD PCM FADER:	AD PCM音源部のマスターフェーダー
MIDI1:	現在演奏中の曲のMIDI1の使用チャンネル数
MIDI1 FADER:	MIDI1のマスターフェーダー
MIDI2:	現在演奏中の曲のMIDI2の使用チャンネル数
MIDI2 FADER:	MIDI2のマスターフェーダー
MIDI3:	現在演奏中の曲のMIDI3の使用チャンネル数
MIDI3 FADER:	MIDI3のマスターフェーダー
SYSTEM:	Z-MUSICのバージョン番号
TIMER:	使用タイマ
最上部の5つのスライダー	それぞれFM,AD PCM,MIDI1～3のマスターフェーダーを表している。

13．4．1．2．トラックパラメータ

コモンパラメータの下には、各トラックごとのワークが表示される。一度に表示できるトラック数は、X680x0の画面の広さの関係から10トラックまでとなっているが、カーソルキーの上下を使ってスクロールさせれば任意のトラック番号のワークを参照することができる。
以下にトラックパラメータ表示部のパラメーター一覧を示す。

オレンジ色の数字:	トラック番号
赤、黄色、水色の文字:	トラックに割り当てられているデバイス名とチャンネル番号
STATUS:	トラックステータス
8桁の16進数:	現在実行しているZMDのアドレス 続いて ZMD,音符の種類,ゲートタイム,ステップタイム,ベロシティが表示されている。 音符がポルタメント指定の場合、さらにポルタメント先の音符が表示される
CH.FADER:	チャンネルフェーダー
TIMBRE:	音色番号(MIDI部はバンク番号-音色番号と表示される)
VOLUME:	音量
VELOCITY:	ベロシティ(FM音源部とAD PCM音源部は音量値と常に同値)
DETUNE:	ディチューン
PITCH:	オートピッチベンダー
AFTER:	アフタータッチシーケンスのアフタータッチ値
HOLD:	ダンパー(サスティン)ペダル
PAN:	パンポット
ARCC1～4:	ARCCの振幅
WAVE:	ARCCの波形種別
VARIANT:	ARCCの振幅の変化量
CTRL:	ARCCでコントロールしているコントロール番号 FM音源部でワウワウやアンプリチュードモジュレーションを行っている場合には操作対象のオペレータ番号が緑色で表示される。
VIBRATO:	ビブラートの振幅
WAVE:	ビブラートの波形種別
VARIANT:	ビブラートの振幅の変化量
VELOCITY.SEQ:	ベロシティシーケンスのときのベロシティ値
WAVE:	ベロシティシーケンスの波形種別
VARIANT:	ベロシティシーケンスの振幅の変化量
鍵盤左の5つのスライダー	ARCC1～4の振幅とピッチの変化を表す。 ARCC1 ARCC2 PITCH ARCC3 ARCC4 のように配列している。
鍵盤の右の○	パンポットつまみ
パンポットつまみの右のゲージ	ダンパーペダルの踏み具合。 オレンジのゲージが上に行けば行くほど深い踏み込み
パンポットつまみの右の小さい●	赤:ノートオフメッセージなし 緑:ノートオフメッセージあり
ダンパーゲージの右のスライダー	チャンネルフェーダー
鍵盤の右のゲージ	アタックレベル キーオンすると立ち上がる。タイやスラーなどの手法で発音、音程変化しても立ち上がらない。

13．4．2．キー操作

ZSV.Rでは、

```
A>ZP3.R -K
```

によってZP3.Rを常駐させてから使用可能となるキー操作はすべて装備しており、環境変数zp3_keyctrlに記載されているキー配置情報も有効となる。環境変数zp3_keyctrlについてはMEASURE7を参照のこと。
以下に操作キー一覧を示す

●ファンクションキー部

[F1]～[F10]	表示されている10トラックをそれぞれマスク/マスク解除する。たとえばトラック3-12が表示されていると[F5]ではトラック7が操作対象となる

●カーソルキー部

[↑]	上へスクロール
[↓]	下へスクロール
[←]	左へスクロール
[→]	右へスクロール
[HOME]	マスク反転
[DEL]	マスク全解除
[ROLL UP]	下へ10トラック分スクロール
[ROLL DOWN]	上へ10トラック分スクロール

●フルキー部

[P]	パンポット表示モード変更(0-127←→L63-0-R63)
[,]	振幅表示モード変更(最大振幅←→振幅実行値)
[F]	FMの全チャンネルマスク/マスク解除
[B]	BANKの表示モードの設定 (14bit,上位7bitのみ,7bit:7bit(デフォルト))
[SPACE]	ステータス表示の一時停止(ポーズ)

●テンキー部

[9]	AD PCMの全チャンネルマスク/マスク解除
[/]	MIDIインタフェイス1の全チャンネルマスク/マスク解除
[*]	MIDIインタフェイス2の全チャンネルマスク/マスク解除
[-]	MIDIインタフェイス3の全チャンネルマスク/マスク解除
[+]	MIDIインタフェイス4の全チャンネルマスク/マスク解除

MEASURE 14

効果音モードと外部プログラムとの同期

ここでは、Z-MUSICの特殊機能である「効果音モード」と「外部プログラムとの同期」についての解説を行います。本章を理解するにはMPU680x0の機械語の知識が必要です。
通常の音楽演奏のみを行なうユーザーは読み飛ばして構いません。
「効果音モード」の解説はオリジナルゲームプログラムで実際に使用することを想定して具体的に解説します。

14．1． 効果音モード

14．1．1． ZMSC3．Xの組み込み

14．1．1．1．効果音トラックの確保

効果音モードを利用するにはZMSC3.Xを常駐させるときに、コマンドオプション'-F'を設定し、効果音用のトラックを確保しなければならない。効果音用のトラックを確保しない場合、効果音モードを利用した効果音の演奏は一切行えない。

－Fn
効果音モード用にトラックをn本確保する。効果音モードを利用するときは、1～65535を設定する。デフォルトではn=0で「効果音モードを利用しない」設定となる。効果音トラックに何トラック割り当てても音楽演奏のトラック数の最大数には影響ない。
例

```
A>ZMSC3 -F4        (効果音トラックに4トラック確保)
```

14．1．1．2．組み込みメッセージの制御

通常、ZMSC3.Xを常駐させると、Z-MUSICのロゴやバージョン番号、認識したインタフェイス等の情報が画面に出力されるが、コマンドオプション'-G'を設定すると、これらを一切表示しないようにすることができる。アプリケーションを起動する際の画面バランスに気を使う場合は設定するとよい。

－G
ZMSC3.X起動時のロゴ表示や常駐を報告するメッセージを一切表示しない。

14．1．1．3．初期化処理の制御

Z-MUSICでは通常、新たに演奏データを演奏する際、MIDI機器に初期化メッセージを送信したりFM音源、AD PCM音源の初期化処理を実行している。この初期化処理は新たに演奏する曲の都合を無視したものであり、かなり余計な処理を含んでいる。たとえばMIDI音源の10以下(1～10)のチャンネルしか使用していない曲を演奏しようというときも11以上(11～16)のチャンネルの初期化メッセージも送信してしまう。これはゲームなどのBGMの切り替え時においてゲーム画面のスクロールの一時停止といった目に見える症状になってしまう場合がある。

ZMSC3.X常駐時にコマンドオプション'-N'を設定すると、以後この初期化処理を必要最低限に省略することができる。ただし全デバイスに対して、初期化メッセージ送信処理を省いてしまうため、演奏データ(ZMD)側に必要な初期化メッセージを盛り込む必要が出てくる（後述）。

－N
初期化処理を簡略化する。この設定を行ってZMSC3.Xを常駐させると以後、演奏開始時に最低限の初期化処理しか行わなくなる。ゲームなどのBGMを機械語レベルで演奏制御する場合に、曲の変り目の処理を軽くする事ができる。
通常は設定する必要はない。

14．1．1．4．ZPDの組み込み

ゲームなどにおいて、用いるすべてのAD PCM音を1つのZPDにまとめている場合はコマンドオプション'-Z'を使用してZMSC3.X常駐時に一緒に組み込んでしまうとよい。

－Z filename
filename.ZPDで与えられるAD PCM BLOCK DATA「ZPD」をZMSC3.X常駐時に読み込む。ファイル名拡張子を省略すると自動的に'.ZPD'が添付される。
例

```
A>ZMSC3.X -Zstandard.zpd
```

14．1．2．プログラムに負担をかけない演奏データ

演奏データ制作者はゲームプログラムに負担をかけないように心掛けて演奏データを作成するよう留意すべきである。また、曲の切り替りに時間がかからないように工夫しないと、ゲームの進行を一瞬停止させてしまう恐れがある。

たとえばスクロールシューティングゲームでステージBGMからボス登場のBGMに切り替るときに、スクロールが一瞬停止してしまってはスピード感を損ねてしまう。

そこで、まず大前提となるのは利用する演奏データ形式はZMDにべきであるという点。ZMSでは演奏毎にコンパイラを呼び出してZMDへ変換する必要があるため、ゲームプログラムへの負担は免れないからだ。

ここでは、ゲームプログラムに負担をかけないZMDの作成方法、曲の切り換えをできるだけ速く行えるZMDの作成方法について述べる。

14．1．2．1．ZMSコマンドは極力使わない

FM音源音色の定義コマンドやMIDIエクスクルーシブメッセージ送信コマンドなどのトラックに依存しない、いわゆる「共通コマンド(ZMSコマンド)」は割り込みで処理されないために実行にはある程度のまとまった時間を要する。したがって、曲の切り替え時などにウェイトになってしまう可能性がある。そこでゲームのBGMにはなるべくZMSコマンドは用いないほうがよい。

MMLコマンドは割り込み処理されるので、ゲームプログラムの実行を完全に止めてしまうことはない。そのため、もしMMLに置き換えられるならばZMSコマンドは用いずに、MMLで代用したほうがよい。たとえば、ZMSコマンド、

```
.ROLAND_EXCLUSIVE
(Xn1,n2,n3,…,ni)
```

などはMMLの、

```
X
@X
```

に置き換えられる。

どうしても置き換わらない、

FM音源音色登録

などは後述する方法で解決する。

なお、曲データである以上必要最低限となるものとして、チャンネルア

サインZMSコマンド、
(A dev,trk,trkv,trkf,trks,trkm,cmnt)
.ASSIGN trk {dev,trkv,trkf,trks,trkm,cmnt}
があるが、これはコンパイル時に消化されてZMDには出力されないので、用いてもゲームプログラム側の負担にはならない。

14．1．2．2．音色登録

演奏データにはさまざまな音色が使用されるが、この音色データを１曲ごとに持たせて演奏の開始時にいちいちZ-MUSICや音源側に登録していたのでは曲の切り替りが遅くなってしまう（アクションゲームのような極端な高速切り替えが要求されていない、RPGやアドベンチャーゲームなら別に構わないであろうが）。

Z-MUSICではFM音源32768音色分(No.1～No.32768)の専用バッファを持っており、ZMSC3.Xが解除されなければ、この領域はいつまでも保存されている。よって、そのゲーム中に使用されるFM音源音色をあらかじめZMD化しておき、これをZMSC3.X常駐時にコマンドオプション'-S'を用いて設定してしまい、その後演奏させる演奏データ側には一切FM音源音色データを持たせずに、あらかじめ登録した音色のみを使用するようにすれば、演奏ごとの音色登録動作を省くことができる。また効果音データで用いるFM音源音色も一緒に設定してしまうとより効率的である。

MIDI楽器の音色についてもFM音源音色の場合と同様で、そのゲームのBGMで使用する音色データ等をあらかじめMIDI楽器へ送信してしまえばよい(初期化メッセージを受信するとあらかじめ設定した音色までも初期化してしまう機種もあるので注意)。

音色のセットアップZMSファイルの例をリスト１に示す。このリスト１のようなものをZMC.XでコンパイルしてZMD化しておき、
A>ZMSC3.X　-sSETUP.ZMD
のようにしてZMSC3.X常駐とともに組み込んでしまえばよい。

ゲームプログラムが複数のMIDI楽器に対応している場合は、MUSICコンフィギュレーションのような場所で楽器機種をユーザーに選ばせ、その時点であらかじめ用意しておいたその楽器ごとの音色データを転送すればよいだろう。

●リスト1

```
/                効果音ＦＭ音色データ
(v1,0                                            /自機ミサイル
/       AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
        60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0
/       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
        27, 15,  5,  2,  0,  0,  0,  0,  3,  1,  0
        31, 18, 18,  6,  7,  0,  0,  0,  3,  2,  0
        22, 31,  0, 10,  0, 42,  0,  7,  7,  0,  0
        15, 31,  0,  8,  0,  0,  2,  1,  7,  0,  0)
                          :
                          :
                          :
                          :

(v20,0                                           /敵レーザー
/       AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
        58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0
/       AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
        12,  5,  1,  5,  5, 24,  1,  0,  0,  1,  0
        16,  4,  1,  5,  2, 30,  1,  0,  3,  0,  0
        29,  6,  5,  5,  2, 16,  2,  0,  3,  2,  0
         9,  4, 31,  8, 10,  0,  2,  4,  3,  1,  0)

/                楽器ＦＭ音色データ

/       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME    BASS 1
(@30,   21,  0,  0,  0,  1, 20,  1,  0,  7,  0,  0
        21,  0,  0,  4,  0,  0,  1,  1,  7,  0,  0
        21,  0,  0,  0,  0, 20,  1,  0,  3,  0,  0
        21,  0,  0,  4,  0,  5,  1,  1,  3,  0,  0
/       AL  FB  OM
         4,  5, 15)

                          :
                          :
                          :
                          :
/       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME    BRASS1
(@100,  19, 17,  0,  8,  1, 24,  1,  1,  1,  0,  0
        19, 13,  0,  8, 15, 37,  1,  2,  0,  0,  0
        19, 17,  0,  8,  1, 36,  1,  1,  0,  0,  0
        19,  4,  5,  8,  2,  0,  1,  1,  2,  0,  0
/       AL  FB  OM
         2,  7, 15)
/-----------------------------------------------------
/                ＭＩＤＩ楽器音色

/       Chord (Saw&Squ)

.roland_exclusive 16,22 ={8,0,0
        67, 72, 79, 82, 68, 32, 32, 32, 32, 32
         0,  0,  3,  0
/ ** PARTIAL1 **
        36, 53, 11,  1,  2,  0, 36,  9
         0,  0,  0,  0,  0,  0,  0, 50, 50, 50, 50, 50
        62, 23, 85
       100, 23,  4, 39,  8, 24
       100,  0,  0,  0, 13, 24, 77, 83,100, 89, 76, 72
        94, 74, 91, 12, 27, 12
         0,  0,  0,  0,  0,  9, 43,100,100,100, 99
/ ** PARTIAL2 **
        36, 48, 11,  1,  3,  0,  0,  0
         0,  0,  0,  0,  0,  0,  0, 50, 50, 50, 50, 50
        64, 23, 56
       100, 23,  5,103,  6, 43
       100,  0,  0,  0,  7, 22,100, 98,100, 94, 84, 79
        94, 70, 91, 12, 27, 12
         0,  0,  0,  4, 11, 14, 15,100, 96, 92, 91}

.mt32_patch 1,16 ={2,0,24,52,2,0,1}

/-----------------------------
/       Keyboard (Harpsichord)

.roland_exclusive 16,22 ={8,4,0
        72, 65, 82, 80, 83, 73, 67, 79, 82, 68
         2,  5,  3,  0
/ ** PARTIAL1 **
        36, 46, 16,  1,  0, 43,  0,  7
         0,  0,  0,  0,  0,  0,  0, 50, 50, 50, 50, 50
         0,  0,  0
         0,  0,  3,  0,  7,  0
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0
        87, 77, 91,  6, 27, 12
         3,  0,  0, 23, 60, 83,  0,100, 88, 88, 88
/ ** PARTIAL2 **
        48, 51, 16,  1,  2,  0,100,  7
         0,  0,  0,  0,  0,  0,  0, 50, 50, 50, 50, 50
         0,  0,  0
       100, 16,  9,103,  8,  0
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0
       100, 75, 27, 12, 15,  5
         1,  1,  0, 23, 60, 92,  0,100, 88, 88, 88}

.mt32_patch 2,16 ={2,2,24,50,2,0,1}

                          :
                          :
                          :
```

14．1．2．3．コメント

演奏データのタイトルや作者名を記述するのに使われている.comment文も演奏時には文字列をスキップする動作を強いられるため、これがウェイ

トになりかねないので用いないようにする。

14．1．2．4．初期化

14.1.1.3.で解説したように、ZMSC3.Xを(コマンドオプション'-N'を付けて常駐させて)初期化処理簡略モードで常駐させている場合、各演奏データごとに音源初期化などの設定を盛り込まないと、音源は前に演奏していた演奏データの設定状態のまま次の曲を演奏してしまう。

Z-MUSICでは、(初期化簡略モードでない)通常モードの場合、新たな演奏データの演奏開始直前に以下のような処理を行っている。

●FM音源部

ノイズモードオフ(Y15,0相当)

●AD PCM音源

AD PCMのパンポット、再生周波数の初期化

●MIDI音源部

リセット　オール　コントローラーズ($Bn,$79,$00)
オムニ　モード　オン($Bn,$7D,$00)
モノ　モード　オフ($Bn,$7F,$00)
ローカル　オン($Bn,$7A,$7F)
ファインチューニングニュートラルへ($Bn,$65,$00,$Bn,$64,$01,$Bn,$06,$40,$Bn,$26,$00)
コースチューニングをニュートラルへ($Bn,$65,$00,$Bn,$64,$02,$Bn,$06,$40)
ピッチベンドレンジ＝1オクターブ($Bn,$65,$00,$Bn,$64,$00,$Bn,$06,$0C)
ボリューム＝63($Bn,$07,$3F)
パンポット＝中央($Bn,$0A,$40))
ピッチモジュレーション＝0($Bn,$01,$00)
ダンパーオフ($Bn,$40,00)
ピッチベンドをニュートラルへ($En,$00,$40)
※nはMIDIチャンネルの内部表現($00-$0F)

初期化なしモードと通常モードの違いは、これらの初期化動作が行われるか、行われないかの違いになってくる。

演奏データをゲームプログラムなどに取り込む前に一度、初期化簡略モードで演奏してみるとよい。それで演奏がおかしかい場合、どの初期化処理が足りなかったのかを見極め、上で示した初期化処理をひとつずつピックアップして演奏データ側に盛り込んでいくとよいだろう。

こうしてみるとFM音源とAD PCM音源は初期化なしモードと通常モードの区別を特に意識する必要はないかもしれない。

ところで、MIDI楽器の場合、ローランド系のDTM楽器（CM/SCシリーズなど）では楽器の特定のメモリアドレスにダミーデータを書き込むと本体内の全パートを初期化してくれる便利な機能を持っている。これを使えば上のような初期化メッセージをすべて送らなくても少ない手間で初期化が行える。いくつか例を挙げておく（初期化したあとの楽器の状態については各楽器マニュアルを参照し確認すること）。

●MT32/CM32L/CM64/CM32P/CM500のＬＡ音源部の場合

$7F,$00,$00にダミーデータを書き込む

例

共通コマンド….ROLAND_EXCLUSIVE $10,$16＝{$7F,$00,$00,$00}
MML…@I$41,$10,$16 X$7F,$00,$00,$00

●GS音源系の場合

$40,$00,$7Fにダミーデータを書き込む

例

共通コマンド….ROLAND_EXCLUSIVE $10,$42＝{$40,$00,$7F,$00}
MML…@I$41,$10,$42 X$40,$00,$7F,$00
(GS音源はこの方法で初期化するとベンドレンジが2半音とされてしまう点に注意)

14．1．3．　演奏開始ルーチン

演奏データ(ZMD)をゲームのメインプログラムから演奏するためには、簡単な演奏マネージメントプログラム(つまりは演奏開始ルーチン)が必要となる。さまざまな演奏開始ルーチンが考えられるがサンプルとしてリスト2を示す。

この例では各演奏データがmusic0、music1、music2…とそれぞれ偶数番地に読み込まれているとする。そこでmusic0を演奏したいならばd0.lに0、music1を演奏したいならばd0.lに1を入れ、このルーチンplay_musicをサブルーチンコールするようにする。これで演奏が開始される。

ZMDの演奏にはZMUSIC ver.3.0のファンクションZM_PLAY_ZMDを利用するが、このファンクションの「a1.lは演奏データの先頭8バイトヘッダの次のアドレスを指していなければならない」という仕様から演奏データテーブルの各要素には +8 が加算してある点に注意。詳しくはファンクションZM_PLAY_ZMDの項を参照。

●リスト2

```
        include zmcall.mac

Z_MUSIC macro   func            *ZMUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

play_music:
        * < d0.l=曲番号(0～16383)
        movem.l d0-d2/a0-a1,-(sp)
        add.l   d0,d0
        add.l   d0,d0
        movea.l music_data_tbl(pc,d0.l),a1
        moveq.l #0,d2                           *高速応答モード
        Z_MUSIC #ZM_PLAY_ZMD
        movem.l (sp)+,d0-d2/a0-a1
        rts

music_data_tbl:
        dc.l    music0+8
        dc.l    music1+8
        dc.l    music2+8
        dc.l    music3+8
        dc.l    music4+8
        dc.l    music5+8
        dc.l    music6+8
        dc.l    music7+8
                :
                :
```

14．1．4．効果音データの作成とその活用方法

Z-MUSIC ver.2.0以前では、効果音を演奏させるには、音楽演奏データ側のトラックとチャンネルの関係を把握しておかなければならなかったが、ver.3.0では音楽データと効果音データの演奏はそれぞれ独立した処理系で演奏されるため、特に前準備は必要としなくなっている。FM音源の音楽を演奏中にFM音源の効果音を演奏させても、チャンネル間の競合はZ-MUSIC側で処理されるため、音楽演奏、効果音演奏、双方とも任意のチャンネルを割り当てることができる。

また、テンポに関しても音楽側と効果音側で独立したテンポを設定することができるようになった。

ここでは、効果音データの作成方法とその演奏方法について解説する。

14．1．4．1．効果音データ

効果音データは、
1)　完全な音楽データと同様の形態をとるもの
2)　演奏データトラックのみで構成されるもの
の２通りがある。

1)のタイプは共通コマンド(ZMSコマンド)までも含んだ完全なZMDファイルで、通常のZMSをZ-MUSIC MML コンパイラ(ZMC.X)でZMD化したものに相当する。2)のタイプは前者から演奏データ部分だけを分離したものである。

1)のタイプの演奏はリスト２のZM_PLAY_ZMDをZM_PLAY_ZMD_SEにした流用ルーチンが利用できる。ただし、1)のタイプの演奏には共通コマンド部分の実行などの手間が入るため、効果音演奏まで一瞬のタイムラグが発生する可能性がある。そのため高速応答が要求されるアクションゲームの効果音には向かない。

2)のタイプはZMDのトラック・チャンネル情報テーブル(MEASURE12参照)と実際のトラック演奏データのみで構成される。こちらの効果音データの作成方法を以下に示す。

まず、リスト3のようなZMSがあったとする。

●リスト3

```
(v12,0                                    /音色データは事前に登録しておく
/          AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
           59, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0
/          AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME
           23,  8,  1,  8,  7, 31,  2,  1,  5,  3,  0
           30,  8,  8,  7,  5, 20,  1,  2,  1,  2,  0
           22,  3,  7,  8,  1, 20,  1,  1,  3,  0,  0
           27,  0,  0,  9,  0,  0,  1,  2,  5,  1,  0)

(i)
.assign TR1 {FM8}
(t1)@12 o3 q8 v14 p3 @k0 @m60 @h24 @s3 (a16<a)&a4.&(a16>a)
(p)
```

これをコンパイルしてZMD化するとリスト4のようになる。

●リスト4

```
00000000  1A 5A 6D 75 53 69 63 30 00 00 00 44 00 00 00 76 .ZmuSic0...D...v
00000010  00 00 00 00 00 00 00 D4 00 00 00 00 00 00 00 00 .......ﾔ........
00000020  00 00 00 00 00 00 00 00 00 00 60 00 00 00 00 00 ........... ....
00000030  00 00 00 00 00 00 00 C0 00 78 00 00 00 00 00 41 .......ﾀ.x.....A
00000040  00 00 00 00 00 00 00 00 01 00 00 00 00 00 00 00 ................
00000050  18 00 0B 00 00 80 00 78 FB 00 51 31 12 52 1F 14 .....  x..Q1.R..
00000060  14 00 97 56 5E 5B 08 03 08 00 C1 07 88 40 78 18 ..遊^[....ﾁ..@x.
00000070  57 09 20 20 20 20 20 20 20 20 20 20 20 20 20 20 W.
00000080  20 20 00 00 FF FF 00 00 00 00 00 00 00 00 00 07   ..............
00000090  00 00 00 14 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000A0  00 00 00 00 00 00 00 00 C7 00 0B 90 8E A0 40 B9 ........ﾇ..錘 @ｹ
000000B0  00 00 E2 01 01 00 3C E4 00 01 00 18 E3 01 00 06 ......<.........
000000C0  84 39 45 0C 80 00 80 45 48 80 00 80 84 45 39 0C .9E.  .E H  .  E9.
000000D0  0C 80 FF FF                                     ... .
```

トラック・チャンネル情報テーブルはオフセットz_trk_offset(MEASURE12参照)で表される位置にあるので、リスト4の場合は

$00000076+$10=$00000086

で、先頭から$00000086のところにあることになる。2)の方式の効果音データではこれより前のデータが不要であるため削除してしまう。これがリスト5である。

●リスト5

```
00000000
00000010
00000020
00000030                      削除してしまう
00000040
00000050
00000060
00000070
00000080                    00 00 00 00 00 00 00 00 00 07   ..............
00000090  00 00 00 14 00 00 00 00 00 00 00 00 00 00 00 00 ................
000000A0  00 00 00 00 00 00 00 00 C7 00 0B 90 8E A0 40 B9 ........ﾇ..錘 @ｹ
000000B0  00 00 E2 01 01 00 3C E4 00 01 00 18 E3 01 00 06 ......<.........
000000C0  84 39 45 0C 80 00 80 45 48 80 00 80 84 45 39 0C .9E.  .E H  .  E9.
000000D0  0C 80 FF FF                                     ... .
```

これをプログラムソース形式に変換したものが、リスト6。これには参考までにトラックチャンネル情報テーブルのキャプションを添付してある。

●リスト6

```
        .dc.b   $00,$00             *ti_n_of_tracks(演奏トラック数-1)
        .dc.b   $00                 *ti_track_stat
        .dc.b   $00                 *ti_track_mode
        .dc.b   $00                 *ti_trkfrq
        .dc.b   $00                 *dummy
        .dc.b   $00,$00             *ti_type
                                    *(デバイスタイプ:0,1,$8000～$8003)
        .dc.b   $00,$07             *ti_ch(チャンネル番号:0-15)
        .dc.b   $00,$00,$00,$14     *ti_play_data(演奏データまでの
                                    *オフセットアドレス)
        .dc.b   $00,$00,$00,$00     *ti_extra_info(トラック追加情報
                                    *までのオフセット)
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$0b,$90,$8e,$a0,$40,$b9  *演奏データ
        .dc.b   $00,$00,$e2,$01,$01,$00,$3c,$e4
        .dc.b   $00,$01,$00,$18,$e3,$01,$00,$06
        .dc.b   $84,$39,$45,$0c,$80,$00,$80,$45
        .dc.b   $48,$80,$00,$80,$84,$45,$39,$0c
        .dc.b   $0c,$80,$ff,$ff
```

リスト7のように複数トラックで構成される効果音データも同様にして作成可能である。これをコンパイルし同様の手続きで変換してやるとリスト8のようになる。

●リスト7

```
(i)
.assign tr1 {FM7}
.assign tr2 {FM8}
(t1)@7 o7 q8 v15 p3 @k0 @l1e<g>>cd<<e>(f8<f)p0
(t2)@4 o7 q8 v12 p3 @k0 L32(c>c)&(c<e)r1
(p)
```

●リスト8

```
        .dc.b   $00,$01             *ti_n_of_tracks(演奏トラック数-1)
        .dc.b   $00                 *ti_track_stat
        .dc.b   $00                 *ti_track_mode
        .dc.b   $00                 *ti_trkfrq
        .dc.b   $00                 *dummy
        .dc.b   $00,$00             *ti_type
                                    *(デバイスタイプ:0,1,$8000～$8003)
        .dc.b   $00,$06             *ti_ch(チャンネル番号:0-15)
        .dc.b   $00,$00,$00,$24     *ti_play_data(演奏データまでのオフ
                                    *セットアドレス)
        .dc.b   $00,$00,$00,$10     *ti_extra_info(トラック追加情報
                                    *までのオフセットアドレス)
        .dc.b   $00                 *ti_track_stat
        .dc.b   $00                 *ti_track_mode
        .dc.b   $00                 *ti_trkfrq
        .dc.b   $00                 *dummy
        .dc.b   $00,$00             *ti_type
                                    *(デバイスタイプ:0,1,$8000～$8003)
        .dc.b   $00,$07             *ti_ch(チャンネル番号:0-15)
        .dc.b   $00,$00,$00,$4c     *ti_play_data(演奏データまでの
                                    *オフセットアドレス)
        .dc.b   $00,$00,$00,$38     *ti_extra_info(トラック追加情報
                                    *までのオフセットアドレス)
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$06,$90,$8f,$a0,$40,$b9  *第1トラックの演奏
                                                 *データ
        .dc.b   $00,$00,$64,$01,$01,$80,$73,$01
        .dc.b   $01,$80,$54,$01,$01,$80,$56,$01
        .dc.b   $01,$80,$70,$01,$01,$80,$84,$65
        .dc.b   $71,$18,$18,$80,$a0,$80,$ff,$ff
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$03,$90,$8c,$a0,$40,$b9  *第2トラックの演奏
                                                 *データ
        .dc.b   $00,$00,$84,$60,$54,$06,$80,$00
        .dc.b   $80,$84,$54,$64,$06,$06,$80,$80
        .dc.b   $80,$c0,$80,$c0,$ff,$ff,$ff,$ff
```

14．1．4．2．効果音演奏形式

効果音を鳴らす場合には2通りの形式が存在する。

ひとつは効果音が鳴るチャンネルをあらかじめ空けておく形式である。つまり効果音を演奏するチャンネルをあらかじめ決めておき、これ以外のチャンネルを使用して音楽(BGM)側のデータを作成するわけである。たとえば、FM音源8チャンネル目を効果音チャンネルと割り当てた場合、1チャンネルから7チャンネルまでをBGM専用チャンネルとしてBGMを作成する。

2つ目はBGMを演奏しているチャンネルに割り込んで効果音を演奏する形式である。たとえばFM音源の1から8チャンネルすべてのチャンネルでBGMを演奏し、そのうち8チャンネルが状況に応じて効果音演奏に切り換わったりする、という場合である。この場合では、効果音が頻繁に演奏されると、8チャンネル目はほとんど効果音しか鳴らなくなってしまい、BGMの1声分が欠けて聞こえることになる。よって、この手法で効果音を鳴らす場合、効果音が被るチャンネルには、BGMの重要パート(メロディやベース等)を割り当てないようにすべきだろう。

14．1．4．3．効果音の演奏

リスト9は効果音の演奏マネージメントプログラムの一例である（リスト9中のサンプル効果音は、それぞれリスト6、リスト8とまったく同じもの)。

リスト9中のse_play1はd0.lに鳴らしたい効果音番号を代入して呼び出すと対応する効果音が演奏されるルーチンである。

●リスト9

```
        .includezmcall.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

se_play1:
        * < d0.l=sound number(0-32767)
        * - all
        movem.l d0-d1/a0-a1,-(sp)
        add.w   d0,d0
        move.w  se_tbl(pc,d0.w),d0
        lea     se_tbl(pc,d0.w),a1
        Z_MUSIC #ZM_SE_PLAY
        movem.l (sp)+,d0-d1/a0-a1
        rts

se_play2:
        * < d0.l=sound number
        * < d1.l=ch number
        * - all
        movem.l d0-d2/a0-a1,-(sp)
        add.w   d0,d0
        move.w  se_tbl(pc,d0.w),d0
        lea     se_tbl(pc,d0.w),a1
        moveq.l #0,d2
        move.w  (a1),d0
@@:
        move.l  d1,6(a1,d2.l)   *デバイスチャンネルセット
        addq.w  #1,d1           *チャンネルのオーバーフローは
                                *ノーチェック
        add.l   #16,d2
        dbra    d0,@b
        Z_MUSIC #ZM_SE_PLAY
        movem.l (sp)+,d0-d2/a0-a1
        rts

se_tbl:
        dc.w    se0-se_tbl              *$00
        dc.w    se1-se_tbl              *$01
        dc.w    se2-se_tbl              *$02
        dc.w    se3-se_tbl              *$03
        dc.w    se4-se_tbl              *$04
                        :
                        :
                        :

se0:
        .dc.b   $00,$00                 *ti_n_of_tracks
                                        *       (演奏トラック数-1)
        .dc.b   $00                     *ti_track_stat
        .dc.b   $00                     *ti_track_mode
        .dc.b   $00                     *ti_trkfrq
        .dc.b   $00                     *dummy
        .dc.b   $00,$00                 *ti_type(ﾃﾞﾊﾞｲｽﾀｲﾌﾟ:
                                        *       0,1,$8000～$8003)
        .dc.b   $00,$07                 *ti_ch(ﾁｬﾝﾈﾙ番号:0-15)
        .dc.b   $00,$00,$00,$14         *ti_play_data(演奏データ
                                        *までのオフセットアドレス)
        .dc.b   $00,$00,$00,$00         *ti_extra_info
                                        *(ﾄﾗｯｸ追加情報までの
                                        *ｵﾌｾｯﾄｱﾄﾞﾚｽ)
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$0b,$90,$8e,$a0,$40,$b9 *演奏データ
        .dc.b   $00,$00,$e2,$01,$01,$00,$3c,$e4
        .dc.b   $00,$01,$00,$18,$e3,$01,$00,$06
        .dc.b   $84,$39,$45,$0c,$80,$00,$80,$45
        .dc.b   $48,$80,$00,$80,$84,$45,$39,$0c
        .dc.b   $0c,$80,$ff,$ff
        .even
se1:
        .dc.b   $00,$01                 *ti_n_of_tracks
                                        *       (演奏トラック数-1)
        .dc.b   $00                     *ti_track_stat
        .dc.b   $00                     *ti_track_mode
        .dc.b   $00                     *ti_trkfrq
        .dc.b   $00                     *dummy
        .dc.b   $00,$00                 *ti_type(デバイスタイプ:
                                        *       0,1,$8000～$8003)
        .dc.b   $00,$06                 *ti_ch(チャンネル番号:0-15)
        .dc.b   $00,$00,$00,$24         *ti_play_data(演奏データ
                                        *までのオフセットアドレス)
        .dc.b   $00,$00,$00,$10         *ti_extra_info
                                        *(ﾄﾗｯｸ追加情報までの
                                        *ｵﾌｾｯﾄｱﾄﾞﾚｽ)
        .dc.b   $00                     *ti_track_stat
        .dc.b   $00                     *ti_track_mode
        .dc.b   $00                     *ti_trkfrq
        .dc.b   $00                     *dummy
        .dc.b   $00,$00                 *ti_type(デバイスタイプ:
                                        *  0,1,$8000～$8003)
        .dc.b   $00,$07                 *ti_ch(チャンネル番号:0-15)
        .dc.b   $00,$00,$00,$4c         *ti_play_data(演奏データ
                                        *までのオフセットアドレス)
        .dc.b   $00,$00,$00,$38         *ti_extra_info(トラック追
                                        *加情報までのオフセット)
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$06,$90,$8f,$a0,$40,$b9 *第1トラックの
        .dc.b   $00,$00,$64,$01,$01,$80,$73,$01 *演奏データ
        .dc.b   $01,$80,$54,$01,$01,$80,$56,$01
        .dc.b   $01,$80,$70,$01,$01,$80,$84,$65
        .dc.b   $71,$18,$18,$80,$a0,$80,$ff,$ff
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $00,$00,$00,$00,$00,$00,$00,$00
        .dc.b   $c7,$00,$03,$90,$8c,$a0,$40,$b9 *第2トラックの
        .dc.b   $00,$00,$84,$60,$54,$06,$80,$00 *演奏データ
        .dc.b   $80,$84,$54,$64,$06,$06,$80,$80
        .dc.b   $80,$c0,$80,$c0,$ff,$ff,$ff,$ff
        .even
                        :
                        :
```

se_play1に対してd0.l=0としてコールすると効果音se0がFM音源第8チャンネルで演奏される。なぜFM音源のチャンネル8なのかはse0のキャプションti_type,ti_chを見ればわかる。もし、同内容の効果音を別のチャンネルで演奏したい場合は、この部分を効果音演奏ルーチン側で書き換えればよい(演奏デバイス、チャンネルを変えたZMDを別途用意しても構わないが、こちらの方法の方がメモリを節約できる)。リスト9のse_play2はその一例である。d0.lに効果音番号、d1.lに演奏デバイス、チャンネルをパラ

メータとして与える。se_play2では効果音データのアドレスを得たあと、そのアドレス+6に演奏デバイス・チャンネルを書き込んでいる。+6というオフセットはse0の演奏デバイス・チャンネル情報がこの位置にあることからきている(キャプションti_type,ti_ch)。

そしてd1.lに代入されている演奏デバイスチャンネルのチャンネルの方を+1し、d2.lを+16して、このあとループ制御を行っている。このループ制御によりルーチンは演奏する効果音の構成トラック数の回数分ループするようになっている。これは効果音データが複数トラックで構成されている場合に対応した部分である。たとえば、se1の場合ならば「se1のアドレス+6」と「se1のアドレス+22」に対しても演奏デバイスチャンネルを書き込む必要があるが、このループ制御でこれを行わせている。効果音演奏チャンネルに対して+1しているのは、複数トラックから構成される効果音の各トラックに1ずつ昇順に別々のチャンネルを割り当てる目的で、また、d2.lに対する+16は次のトラックチャンネル情報テーブルをポイントする目的でオフセット加算を行っている。

このように効果音のデータを直接書き換える手法を応用するとさまざまな効果を実現することができる。ここで示した例は、演奏デバイスチャンネルを書き換える程度のものだが、効果音の演奏データそのものを直接書き換えたりすればいっそう応用の幅が広がる。たとえば3Dゲームならば、効果音の演奏データ部分のパンポットZMDや音量ZMDのパラメータを、その3次元座標に合せて直接書き替えて演奏させてやれば疑似立体音響のようなものも実現可能だ。

14.1.4.4. AD PCM音の効果音

AD PCM音の効果音も、これまでに解説してきた方法で演奏できるが、Z-MUSIC ver.3.0ではAD PCM効果音専用のファンクションコールが設けてあり、これを利用することでも演奏させることができる。

リスト10がそれらのファンクションを用いたマネージメントプログラムの一例である。

●リスト10

```
        .includezmcall.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

ADPCM_se_play1:                                 *AD PCM効果音
        *任意のアドレスに格納されたAD PCM音を効果音として鳴らす
        *
        * < d0.l=効果音番号
        *
        movem.l d0-d2/d4/a0-a1,-(sp)
        lsl.l   #3,d0
        movem.l adpcm_se_tbl1(pc,d0.w),d1-d2/d4/a1
        Z_MUSIC #ZM_SE_ADPCM1
        movem.l (sp)+,d0-d2/d4/a0-a1
        rts

adpcm_se_tbl1:
*0  ﾃﾞｰﾀ種類(.b)(MPCM.X常駐時のみ有効)
*                                   -1:AD PCM,1:16bitPCM,2:8bitPCM
*1  音量(.b)(MPCM.X常駐時のみ有効)   0～127(64が原音量)
*2  パンポット(.b)                  1:左 2:右 3:中央
*                                   または $80+(0～127)
*3  再生周波数(.b)                  0～4
*4  データのサイズ(.l)
*8  優先レベル(.w)                  0～255 (0がいちばん低い)
*10 演奏チャンネル(.w)(MPCM.X常駐時のみ有効)
*                                   0～15
*12 データアドレス(.l)

        dc.l    $ff_40_02_03,$0000_000f,2212,adpcm_se0
        dc.l    $ff_40_04_03,$0001_000f,3478,adpcm_se1
        dc.l    $ff_40_03_01,$0000_000f,5453,adpcm_se2
        dc.l    $ff_40_04_02,$0001_000f,1401,adpcm_se3
        dc.l    $ff_40_04_03,$0000_000f,8799,adpcm_se4
                        :
                        :
                        :
                        :

ADPCM_se_play2:
        *Z-MUSICに登録したAD PCM音を効果音として鳴らす
        *
        * < d0.l=効果音番号
        *
        movem.l d0-d2/d4/a0,-(sp)
        lsl.l   #3,d0
        movem.l adpcm_se_tbl2(pc,d0.w),d1-d2/d4
        Z_MUSIC #ZM_SE_ADPCM2
        movem.l (sp)+,d0-d2/d4/a0
        rts

adpcm_se_tbl2:
*0  データ種類(.b)(MPCM.X常駐時のみ有効)
*                                   -1:AD PCM,1:16bitPCM,2:8bitPCM
*1  音量(.b)(MPCM.X常駐時のみ有効)   0～127(64が原音量)
*2  パンポット(.b)                  1:左 2:右 3:中央
*                                   または($80+0～127)
*3  再生周波数(.b)                  0～4
*4  優先レベル(.w)                  0～255 (0がいちばん低い)
*6  演奏チャンネル(.w)(MPCM.X常駐時のみ有効)
*                                   0～15
*8  登録トーン/音色番号(.l)         0～32767または$8000+(0～32767)
        dc.l    $ff_40_02_03,$0000_0007,0,0
        dc.l    $ff_40_04_03,$0001_0007,1,0
        dc.l    $ff_40_03_01,$0000_0007,2,0
        dc.l    $ff_40_04_02,$0001_0007,3,0
        dc.l    $ff_40_04_03,$0000_0007,4,0
                        :
                        :
```

まず、リスト中のADPCM_se_play1について。これは任意のアドレスに格納されているAD PCMデータをファンクションZM_SE_ADPCM1を使って、BGMに割り込ませて鳴らすためのサブルーチンである。BGM側でAD PCMをドラムなどで使用している場合はそれを一時停止して効果音を演奏する。効果音が鳴り終ると自動的にBGMの演奏にもどる。

ADPCM_se_play1ではアドレスに配置されたAD PCMデータに便宜的に番号をつけ、d0.lにその番号を代入して呼び出すようになっている。ルーチン後ろにあるadpcm_se_tbl1は各効果音ADPCMデータのデータ属性、再生属性、優先レベル(囲み参照)、再生チャンネル、データサイズ、データアドレスなどを列記した管理テーブルである。

ADPCM_se_play2はファンクションコールZM_SE_ADPCM2を用いて、Z-MUSIC側にすでにZPDとして登録済みのAD PCM音を効果音として鳴らすためのサブルーチンである。d0.lに演奏したい効果音番号を代入してADPCM_se_play2をコールして使用する。ルーチン後ろにあるadpcm_se_tbl2はADPCM_se_play1のadpcm_set_tbl1からデータサイズとデータアドレスを取り去り、代わりに登録トーン/音色番号を書き込んだものになっている。

AD PCM効果音の優先レベル

X680x0のAD PCMは(MPCM.Xなど非常駐時は)単声仕様です。ですから効果音の発音要求は基本的に後着優先、つまり新しく発音要求されたものが優先的に発音されることになっています。しかし、ゲームなどで面と面との繋ぎのイベントの音声メッセージなど、どうしても爆発音などでかき消されたくない場合があります。そこでZ-MUSIC ver.3.0ではAD PCM効果音を発音させるときに優先レベルというものを設定できるようになっています。優先レベルは0～255で設定可能です。実用上は0,1程度で十分でしょう。

以下に優先レベル0,1を用いた例を示します。

発音中の優先レベル	発音要求の優先レベル	結果
0	0	発音要求採択
0	1	発音要求採択
1	0	発音要求却下
1	1	発音要求採択

14．2． 外部プログラムとの同期

Z-MUSIC ver.3.0には外部プログラムと音楽演奏を同期させるためのファンクションコールや機能が装備されている。ここではこれらの機能について解説を行う。

14．2．1． ファンクションZM_OBTAIN_EVENTS

ファンクションZM_OBTAIN_EVENTSは、Z-MUSIC ver.3.0とユーザープログラムを仲立ちするファンクションで、ユーザープログラムで設定した条件を満たすと、ZMUSICからユーザープログラム側へルーチンの実行が移る仕組みになっている。

MEASURE10より抜粋

ファンクション$58 ZM_OBTAIN_EVENTS

機能　各種イベントの取得

引数　d1.w:どのイベントに関する設定を行うかのビットマーカー
　　ビット=1でそのイベントの設定を更新する。
　a1.l:各種イベント発生時のジャンプ先とジャンプ条件が
　　記述された領域。構成は備考参照。
　　-1とするとint omtによらず設定は一切行わない。

戻り値　a0.l=現在の設定を格納した領域のアドレス。構成は備考参照。

備考
```
$00(a1):m_playが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ         d1.wのd0ビット
$04(a1):m_stopが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ         d1.wのd1ビット
$08(a1):m_contが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ         d1.wのd2ビット
$0c(a1):演奏が終了時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ           d1.wのd3ビット
$10(a1):指定ループ回数(0:OFF,1-256)          d1.wのd4ビット
$14(a1):指定回数ループ時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ       d1.wのd5ビット
$18(a1):指定クロック(0:OFF,1～)              d1.wのd6ビット
$1c(a1):演奏開始後指定クロック経過したときのｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ
                                             d1.wのd7ビット
```

d1.wでビット=1としたパラメータのみをa1.lから記述しておく。d1.wでビット=0としたパラメータに関しては記述しない。たとえばd1.w=$81の場合、a1.lからは、

　$00(a1)に「m_playが実行されたときのジャンプ先アドレス」
　$04(a1)に「演奏開始後指定クロック経過時のジャンプ先アドレス」

を記述しておけばよい。

なお、各ジャンプ先のルーチンではレジスタを破壊してはならず(保存する必要がある)、ルーチンの終端は「RTS」命令で終わる必要がある。

具体的には、

・演奏が終了したら指定アドレスへジャンプ
・演奏開始後、指定クロック(絶対音長)でnカウント経過したら指定のアドレスへジャンプ
・演奏をx回繰り返したら指定のアドレスへジャンプ

といった、ユーザープログラムとZ-MUSIC ver.3.0の連係動作を実現できる。

このほか、ファンクションZM_OBTAIN_EVENTSには「演奏開始」「演奏停止」「演奏再開」が実行されたときのイベントをユーザープログラムが取得できる機能がある。

以下にファンクションZM_OBTAIN_EVENTSの使用サンプルを示す。

●リスト11

```
        .include zmcall.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

sample1:
        movem.l d0-d1/a0-a1,-(sp)
        moveq.l #$08,d1                         *d3ビット=1
        lea     user_event(pc),a1
        Z_MUSIC #ZM_OBTAIN_EVENTS
        movem.l (sp)+,d0-d1/a0-a1
        rts

performance_end:
        movem.l d0-d7/a0-a6,-(sp)
                :
                :
                :
        movem.l (sp)+,d0-d7/a0-a6
        rts

user_event:
        dc.l    performance_end
```

これは演奏中の曲が演奏終了した場合performance_endにジャンプするように設定している。実際のイベント処理ルーチンはperformance_endから始まるが、上のサンプルではその内容までは省略している。

サンプルで示しているようにイベント処理ルーチン内で使用するレジスタはすべてそのルーチンエントリで保存し、終了時に復元しなければならない。また、ルーチンからの帰還には「RTS」を使用している点に注意。割り込み処理とは異なるため「RTE」などを使用してはならない。

なお、同様にして、指定回数ループ、イベント処理ルーチンへジャンプ(リスト12)、指定クロック経過後イベント処理ルーチンへジャンプ(リスト13)といったことも行える。

●リスト12

```
        .include zmcall.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

sample2:

        movem.l d0-d1/a0-a1,-(sp)
        moveq.l #$30,d1                         *d4,d5ビット=1
        lea     user_event(pc),a1
        Z_MUSIC #ZM_OBTAIN_EVENTS
        movem.l (sp)+,d0-d1/a0-a1
        rts

loop_completed:
        movem.l d0-d7/a0-a6,-(sp)
                :
                :
                :
        movem.l (sp)+,d0-d7/a0-a6
        rts

user_event:
        dc.l    10                              *ループ条件
        dc.l    loop_completed
```

●リスト13

```
        .include zmcall.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

sample3:
        movem.l d0-d1/a0-a1,-(sp)
        moveq.l #$c0,d1                         *d3ビット=1
        lea     user_event(pc),a1
        Z_MUSIC #ZM_OBTAIN_EVENTS
        movem.l (sp)+,d0-d1/a0-a1
        rts

n_count_passed:
        movem.l d0-d7/a0-a6,-(sp)
                :
                :
```

```
                :
        movem.l (sp)+,d0-d7/a0-a6
        rts

user_event:
        dc.l    480                             *指定クロック
        dc.l    n_count_passed
```

リスト12では10回ループしたらイベント処理ルーチンへ、リスト13では480クロック経過したらイベント処理ルーチンへ飛ぶようになっている。

14．2．2． ユーザー開放ワークを利用する

ZMUSIC ver.3.0は演奏が開始されると、音楽演奏割り込み処理が起動されるたびに汎用ワークzmusic_intがインクリメントされる(音楽割り込み処理の起動は、絶対音長1ごとに発生する)。この値は外部から参照可能であるため、ここを外部プログラムで監視することで、音楽との同期を実現できる。

以下にそのサンプルを示す。

●リスト14

```
        .include zmcall.mac
        .include zm_stat.mac

Z_MUSIC macro   func            *Z-MUSICへのファンクションコール
        moveq.l func,d0
        trap    #3
        endm

        Z_MUSIC #ZM_GET_ZMSC_STATUS
        move.l  zmusic_int(a0),d0
        cmpi.l  #xxxx,d0
        bcs     @f
                :
            条件成立
                :
@@:
        rts
```

ファンクションZM_GET_ZMSC_STATUSはZ-MUSIC ver.3.0の汎用ワークのベースアドレスを得るファンクションで、a0.lにそのアドレスを返す(ソースリストアーカイブに含まれるzm_stat.macをインクルードすることにより、各ワークをオフセット名で指定することができる)。

zmusic_intの内容を目的の経過クロックと比較して、もし指定クロックを過ぎていたならば「条件成立」内を実行する。

14．2．3． ファンクションZM_LOOP_CONTROL

外部プログラムからループ演奏を制御することもできる。

これにはファンクションZM_LOOP_CONTROLを用いる。

MEASURE10より抜粋

ファンクション$59　　ZM_LOOP_CONTROL

機能　ループ制御

引数　d1.l:コントロールモード
　　-1=ループ回数取得
　　0=通常ループ指定
　　1=ループ無視指定

戻り値　d0.l=-1以外:現在のループ回数(1～)
　　d0.l=-1:エラー

d1.l=1として「ループ無視」とすると、演奏中の曲の各トラックがループ終端にきても、ループ処理を実行せずに、ループ命令の後ろの演奏データを演奏しにいく。たとえば、MMLで、

[do] cde [loop] fga

とあるとき、「ループ無視」のときは cde fga と演奏する。

ユーザープログラム側で時間管理を行い、任意のタイミングで「通常ループ」モードと「ループ無視」モードを切り換えることで、演奏の進行を自由に制御できる。

たとえば、

[do] cde [loop] [do] fga [loop]

という演奏データも、通常モードでは fga が演奏されることはないが、無指定モードと通常モードを外部プログラムで随時切り換えてやることで cde を数回、fdeを無限ループ演奏させることが可能となる。

イントロ演奏を外部プログラムの都合がつくまで演奏させておく目的などに使える。2つの違う曲をひとつの演奏データ内に存在させ、ノンストップミックス調に曲を切り換えることもできる。

MEASURE 15

MPCM.X

ここではZ-MUSIC ver.3.0と組み合わせて使用するPCMドライバ「MPCM.X」について解説します。

15．1． MPCM.Xとは

MPCM.XはZ-MUSIC ver.3.0と組み合わせることで、内蔵AD PCM音源の表現力を引き上げることができるAD PCM音源ドライバである。

扱えるデータの種類はAD PCM/8bitPCM/16bitPCMとなっており、すべての種類のPCMデータに対して各機能が利用できる。

MPCMファンクションコールを利用することで、Z-MUSIC ver.3.0以外のユーザーアプリケーションからも簡単に利用することが可能となっている。

ここでは代表的な機能とその仕組みについて解説する。

15．1．1．PCMデータの多重再生

X680x0本体のAD PCM音源では、通常は1音分のAD PCMしか再生できないが、MPCM.XではPCMデータをリアルタイムに合成することで、擬似的に最大25音(演奏用チャンネル16音+効果音用チャンネル8音+IOCS用チャンネル1音)の再生が可能である。ただし、全チャンネルで再生するためにはかなりのCPUパワーが必要となる(リアルタイムにPCMを合成するためには、高速なPCM変換アルゴリズムを用いなければならないが、MPCM.XではPCM8.Xとほぼ同じものを使っている)。

MPCM.Xでは、再生するチャンネルが、音楽演奏用/効果音(IOCS)用と分かれており、音楽演奏用のチャンネルでは、音程・音量変換やPCMデータのループ再生などが行えるが、少し処理が重くなる。効果音(IOCS)用のチャンネルでは音程・音量変換やループ再生などはできないが、その分処理は軽くなっている。用途によって使い分けるとよい。

15．1．2．音程のリアルタイム変換

MPCM.Xでは、PCMデータの音程をリアルタイムに変換しながら再生を行うことができる。つまり、PCMデータをひとつだけ登録しておけば、あとから自由な音程でそのデータを再生することができる機能を持っている。これまでのように、音程ごとのPCMデータを用意しておく必要はない。また、再生中にも音程を変化させられるため、ピッチモジュレーションなどの音程情緒表現も可能となる。

MPCM.XではPCMの音程変換を、PCMデータの再生周波数を擬似的に変化させてやることで実現している。PCMデータを数個飛ばしで再生してやると、擬似的に再生周波数が上がり、結果として音程もあがる。逆に、同じPCMデータを数回繰り返して再生した場合、再生周波数は下がり、音程も下がる(本来飛ばしたデータや、繰り返したデータの間を補間してやれば、もう少し高音質な音程変換を行うことが可能なはずであるが、処理速度との兼ね合いでMPCM.Xでは行っていない)。

処理速度については、PCMデータの原音程(音程変換をしない場合)での再生は専用ルーチンを使い、ある程度速度を稼いでいる。しかし、微妙に音程をずらした場合はかなり重くなってしまう。またAD PCMデータの場合は、音程を上げる場合、データの先読み処理が発生するため（AD PCMデータは前のデータとの差分でしかなく、飛ばすデータに対してもPCM変換の計算を行う必要がある）、音程を上げれば上げるほど、CPUパワーを必要とする。逆に、下げれば下げるほど処理は軽くなるが、再生時の音質が悪くなってくる。

8bit/16bitPCMデータは、いくら音程を変化させても処理の重さが変わらないため、音程変化を頻繁に行う用途の場合はこれらのPCMデータ形式を用いたほうがよい。

15．1．3．音量のリアルタイム変換

MPCM.XではPCMデータの音量を、リアルタイムに128段階に変化させることができる。Z-MUSIC ver.2.0以前まで標準AD PCM音源ドライバとされてきたPCM8.Xなどよりも細かい音量調節が可能となっている。

音量変換は、指定された音量に対応する定数を、PCMデータに対して掛け算することで実現している。この掛け算に使われる定数の配列（音量マップと呼ぶ）をユーザーが作成登録することもでき、ユーザー独自の音量変化を設定することもできる。たとえば、デフォルトではMPCM.Xの音量変化はPCM8.Xとは異なっているが、音量マップをユーザー側で作成し、これを登録すればPCM8.X互換の音量変化も可能となる。

音量変化には掛け算命令を使用するため、相当量のCPUパワーを消費する。

8，16ビットPCMデータ方式

X680x0に搭載されるAD PCM音源のAD PCMデータとはいわばサンプリングしてできたPCMデータが一種のアルゴリズムで圧縮されていると考えられます。音量の変更や周波数変換、合成処理を行うにはこのAD PCMデータ方式から線形なPCMデータ方式へ変換しなければなりません。MPCM.Xではこの変換処理を割り込み処理でリアルタイムで行っていますが、もしこの処理を省けるならば、MPCM.XのPCM再生処理をかなり軽くすることができるはずです。そこでMPCM.Xでは無圧縮のPCMデータも再生できるようになっています。

16ビットPCMデータ方式は符号付き16ビット整数(－32768～32767)により表現されるデータです。同一音長のAD PCMデータの４倍のデータ長になります。ちょうどZVT.Xでコマンドオプション'-C'により生成されるPCMデータと同じものです。ひとつのデータが２バイト構成であることから必ずデータ長は偶数でなければ正常に鳴りません。

８ビットPCMデータは方式は同様に符号付き８ビット整数(－128～127)により表現されるデータです。同一音長のAD PCMデータの２倍のデータ長になります。

Z-MUSIC ver.3.0およびZPCNV3.Rではこれら８ビットPCMデータ/16ビットPCMデータ専用のZMSコマンドが用意されています(MEASURE4,MEASURE6参照）。

15．1．4．ループ再生

MPCM.Xでは、再生するPCMデータの指定範囲を繰り返し再生することができる。ループをうまく利用することによって、小さいPCMデータで長い再生時間を得ることができる。

15．2． MPCM.X常駐時に設定できるオプション

Z-MUSIC ver.3.0をはじめとするアプリケーションでMPCM.Xを利用する場合、必ず事前にMPCM.Xを常駐させておかなければならない。

以下にMPCM.X常駐時に設定できるコマンドオプションを示す。

●常駐モード選択

－ＦＨ

－FL

X680x0内蔵のAD PCM音源チップ(MSM6258V)の動作周波数を7.8/15.6/31.2kHzから選択することができる。'-FH'は31.2kHz、'-FL'は7.8kHzでMSM6258Vを動作させる。本オプションを設定せずに常駐した場合はデフォルトとして15.6kHzが設定される。

'-FH'はAD PCM音源チップをクロックアップしたマシン上でのみ有効で、高品質のPCM再生を行える。しかし、単位時間に処理しなければいならないPCMデータ量が、デフォルト状態の15.6kHzに比べ2倍になるため、かなりのCPUパワーを持ったマシンでないと正しく動作できない。

一方、'-FL'では音質は劣化するものの処理が軽くなるため、音質よりも、より多くの発音を求める場合には有効なオプションとなる。

15．3． MPCM.X常駐後に設定できるオプション

MPCM.Xを、以下に示す、MPCM.X常駐後に設定できるコマンドオプションを用いて実行すると、既に常駐しているMPCM.Xに対してアクセスすることができる。

●常駐解除

－R

MPCM.Xの常駐を解除する。ほかのアプリケーションにMPCM.Xが占有されている場合は解除を行えない。MPCM.Xを占有しているアプリケーションの一覧は、コマンドオプション'-S'で参照することができる。

●デバッグモード切り換え

－D
－D0
－D1

常駐しているMPCM.Xのデバッグモードを切り替える。'-D0'でデバッグモードOFF(デフォルト)、'-D1'でデバッグモードON、'-D'のみではON/OFF状態を交互に切り替える。

デバッグモードにすると、画面に青い帯が表示される。帯の幅がPCMデータを処理している時間で、この幅の変化で現在の負荷の状態をうかがい知ることができる。通常は設定する必要はない。

このコマンドオプションは常駐後も使用可能となっている。

●MPCM.X常駐ステータスの表示

－S

常駐しているMPCM.Xのステータスを表示する。

15．4．MPCM.Xのファンクションコール

MPCM.XのファンクションはMPCM.X常駐時のみ利用することができる。よって、利用する前には前に必ずMPCM.Xの常駐判定をする必要がある。

MPCM.Xの常駐の判定は、trap #1の指すベクタの手前8バイトが、

'MPCM/???'

になっているかどうかで行う。

???の部分はバージョン番号を表す文字列で、たとえばバージョン番号が0.20のとき、

'MPCM/020'

のようになっている。

例

```
check_MPCM:
*       MPCMの常駐をチェックし、バージョンが0.20より高いかを判定
*
*       判定結果
*       d0.l=-1:常駐していない
*            = 0:常駐している
*            = 1:バージョンが古い
        move.l  a1,-(sp)
        movea.l $84.w,a1
        cmpi.l  #'MPCM',(a1)
        bne     not_kept
        cmpi.l  #'/020',4(a1)  *バージョンが0.20より古いか新しいか
        bcs     ver_too_old    *古いときはエラー
        moveq.l #0,d0
exit_check_MPCM:
        move.l  (sp)+,a1
        rts
not_kept:
        moveq.l #-1,d0
        bra     exit_check_MPCM
ver_too_old:
        moveq.l #1,d0
        bra     exit_check_MPCM
```

15．4．1．MPCM.Xのファンクションコール利用法

MPCM.Xのファンクションコールには、trap #1を使用する。

例

```
    move.w   #n,d0      * n=ファンクションナンバー
    lea.l    ～,a1      * パラメータが必要であれば設定
    move.l   #m,d1      * パラメータが必要であれば設定
    trap     #1
```

■演奏チャンネル用ファンクション

ファンクション　$00xx　　M_KEY_ON

機能： 指定の演奏用チャンネルをキーオンする。
引数： d0.w=$0000 + チャンネル番号(0～15,255)
戻り値： なし
備考： PCMデータ登録($02xx)で登録されたデータを再生する。

ファンクション　$01xx　　M_KEY_OFF

機能： 指定の演奏用チャンネルをキーオフする。
引数： d0.w=$0100 + チャンネル番号(0～15,255)
戻り値： なし
備考： 指定チャンネルのPCMの再生を打ち切る。

ファンクション　$02xx　　M_SET_PCM

機能： 指定の演奏用チャンネルにPCMデータを登録する。
引数： d0.w=$0200 + チャンネル番号(0～15,255)
d1.l=0 (予約)
a1.l=PCM情報テーブルアドレス
戻り値： d0.l≧0 正常終了
<0 エラー
備考： 引数のa1.lに指定する、PCM情報テーブルの内容を以下に示す。

$0000(a1).b=PCM種類
(-1:AD PCM 0:データなし 1:16bitPCM 2:8bitPCM)
$0001(a1).b=PCMのオリジナルノート番号
(0～127、負の値で原音程固定)
$0002(a1).b=0(予約)
$0003(a1).b=0(予約)
$0004(a1).l=PCMアドレス(絶対番地)
$0008(a1).l=PCMの長さ
$000c(a1).l=ループ開始点のデータ先頭からのオフセット
$0010(a1).l=ループ終了点のデータ先頭からのオフセット
$0014(a1).l=ループ回数(0で無限ループ、通常は1)

PCMの長さ(+$0008)は、PCMの種類に関係なく、バイト単位で指定する。したがって、16bitPCMの場合は必ず2の倍数でなければならない。同様に、各種ループオフセット(+$000c,+$0010)も、バイト単位で指定する。各種オフセットは、PCMデータの先頭からのバイト数で表す。

オリジナルノート番号(+$0001)は、そのPCMの原音程を、音階のどこに割り付けるかを指定する。たとえば、オリジナルノート番号を10として、音程に(10+12)*64+0(ファンクション $04xxを参照)を指定すると、1オクターブ上の音程で再生される。

16bitPCM登録時に、アドレス、長さ、ループポインタのいずれかが奇数である場合などは、エラーとなる。

このファンクションでPCMデータを登録しただけでは、データの再生は行われない。再生するためには、ファンクション $00xx(PCM KEY ON)を呼ぶ必要がある。

ファンクション　$03xx　　M_SET_FRQ

機能： 指定の演奏用チャンネルのPCM再生周波数を設定する。
引数： d0.w=$0300 + チャンネル番号(0～15,255)
d1.w=再生周波数

0＝ 3.9kHz
1＝ 5.2kHz
2＝ 7.8kHz
3＝10.4kHz
4＝15.6kHz
5＝20.8kHz
6＝31.2kHz

戻り値：d0.l≧0 正常終了
　　　　　<0 エラー

備考：このファンクションで設定する再生周波数と、再生時の音程は完全に独立している。たとえば、元が15.6kHzのデータを、原音程、再生周波数7.8kHzとして再生すると、7.8kHzで再生される。
　d1.w＝5,6は、AD PCMをクロックアップしているマシンに対応したものだが、クロックアップしていないマシンでも、擬似的に再生する。

ファンクション　$04xx　　　M_SET_PITCH

機能：　指定の演奏用チャンネルのPCM再生音程を設定する。
引数：　d0.w＝$0400 + チャンネル番号(0～15,255)
　　　　d1.w＝音程
　　　　　0～127(ノート番号)*64+0～64(ディチューン)
　　　　　FM音源とほぼ同じ考え(4倍はしなくてよい)
戻り値：なし
備考：d1.wのノート番号は、±1で約半音分音程が変化する。ディチューンはその間をもっと細かく変化させたい場合に指定する（半音の間が、64段階に分かれているイメージである）。
　ファンクション$00xx(PCM KEY ON)で発音したあとも、任意のタイミングで再設定可能である。

ファンクション　$05xx　　　M_SET_VOL

機能：　指定の演奏用チャンネルのPCM再生音量を設定する。
引数：　d0.w＝$0500 + チャンネル番号(0～15,255)
　　　　d1.b＝音量(0-127)
戻り値：なし
備考：d1.wは、音量マップ(ファンクション$8005参照)上のどの値を音量変換に使用するかを決定している。実際の音量は、音量マップ上の値をPCMデータに掛け算することで決定している。

ファンクション　$06xx　　　M_SET_PAN

機能：　指定の演奏用チャンネルのパンポットを設定する。
引数：　d0.w＝$0600 + チャンネル番号(0～15,255)
　　　　d1.b＝PAN
　　　　　0:無音 1:左 2:右 3:中央 (3段階指定)
　　　　　$80 + 0(左)～64(中央)～127(右) (128段階指定)
戻り値：なし
備考：128段階指定の場合、現在のところ、0～31が左、32～95が中央、96～127が右出力となる。各チャンネル独立しておらず、最新の設定に全チャンネルが影響される。

ファンクション　$07xx　　　M_CHG_PCMKIND

機能：　指定の演奏用チャンネルのPCMデータの種類を変更する。
引数：　d0.w＝$0700 + チャンネル番号(0～15,255)
　　　　d1.b＝PCM 種類 (-1:AD PCM 0:なし 1:16bitPCM 2:8bitPCM)
戻り値：なし
備考：ファンクション$02xxで登録したPCMデータの種類のみを変更する。通常は$02xxのほうで指定すること。

■効果音チャンネル用ファンクション

ファンクション　$10xx　　　M_EFCT_OUT

機能：　指定の効果音用チャンネルでPCMを再生する。
引数：　d0.w＝$1000 + 効果音チャンネル番号(0～7,255, $e0)
　　　　d1.l＝PCM種類
　　　　(-1:AD PCM 0:なし 1:16bitPCM 2:8bitPCM)*$1000000
　　　　　＋音量(現在は64のみ指定可)*$10000
　　　　　＋再生周波数
　　　　(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz
　　　　　　　　　　5:20.4kHz 6:31.2kHz) * $0100
　　　　　＋PAN(0-3 / $80+0-127)
　　　　d2.l＝PCM長さ
　　　　a1.l＝PCMアドレス

戻り値：d0.l≧0 正常終了
　　　　　<0 エラー
備考：基本的にIOCS _ADPCMOUTとほぼ同じ引数仕様になっているが、d1.lの上位16ビットの仕様が拡張されている。
　音量は無視されるが、将来のために64($40)として呼び出すこと。それ以外の値で呼んだ場合は将来動作を保証できない。
　また、チャンネル番号に$e0を指定した場合、効果音チャンネルの先頭からファンクション$8005で指定したチャンネル数分、空きを探して再生する。全部ふさがっていた場合、再生残り部分がいちばん短いチャンネルで再生される。この場合、戻り値は実際に割り当てた効果音チャンネル番号を返す(0～7)。
　このファンクションで再生を行った場合、消音は必ずファンクション$13xxを利用して行うこと。

ファンクション　$11xx (reserved)

ファンクション　$12xx (reserved)

ファンクション　$13xx　　　M_EFCT_STOP

機能：　指定の効果音用チャンネルの発音を中止する。
引数：　d0.w＝$1300 + 効果音チャンネル番号(0～7,255)
戻り値：なし

■MPCM.X制御用ファンクション

ファンクション　$8000　　　M_LOCK

機能：　MPCM.Xを常駐解除できないように占有する。
引数：　d0.w＝$8000
　　　　a1.l＝アプリケーション名を指す文字列アドレス
戻り値：d0.l＝0 占有できた
　　　　　<0 占有できなかった
備考：MPCM.Xの常駐を前提に常駐するアプリケーションは、このファンクションで必ずMPCM.Xを占有すること。同時に占有できる数は32個まで。a1.lに指定するアプリケーション名は、コマンドオプション'-S'で表示されるものである。文字列の終端には0.bを置くこと。名前の長さは終端の0.bを含み32byteまでとする。それ以降は無視される。また、アプリケーション名の省略は行ってはならない。

ファンクション　$8001　　　M_FREE

機能：　MPCM.Xの占有解除。
引数：　d0.w＝$8001
　　　　a1.l＝アプリケーション名の文字列アドレス
戻り値：d0.l＝0 占有解除できた
　　　　　<0 占有されていなかった
備考：ファンクション$8000でMPCM.Xを占有したアプリケーションは、必ず終了時にこのファンクションで占有解除を行うこと。a1.lに指定するアプリケーション名は、ファンクション$8000で設定したものと同じでなければならない。

ファンクション　$8002　　　M_INIT

機能：　MPCM.Xの初期化。
引数：　d0.w＝$8002
戻り値：なし
備考：すべての設定値を初期化してしまうので、呼出には注意。音源ドライバなどで利用するなら、常駐時に１回呼ぶだけでよい。

ファンクション　$8003　　　M_SET_MASK

機能：　MPCM.XのDMA割り込み中のMPU/MFPの割り込みマスクの設定。
引数：　d0.w＝$8003
　　　　d1.l＝マスクデータ
　　　　　≧0 割り込みマスク設定
　　　　　<0 割り込みマスク取得
戻り値：d0.l＝変更前の割り込みマスク
　　　　　関係ないビット(31～19)は0になる。
備考：d0.lの内訳を以下に説明する。
　ビット18～16 : MPU 割り込みレベル(0～7 推奨3以上)
　　　15～ 8 : MFP 割り込みマスクA (IMRA)
　　　　7～ 0 : MFP 割り込みマスクB (IMRB)
　ビット15～0では、1で割り込み前と同じ状態になる。
　(つまり、MFPのIMRA,IMRBとANDをとる)

割り込みについてよく理解してから使用すること。
デフォルトは %00000000_00000011_11011111_00000000 である。

ファンクション　$8004　　　M_SET_MODE

機能：　MPCM.Xの動作モードの設定。
引数：　d0.w=$8004
　　　　d1.l=設定データ
　　　　　≧0 モード設定
　　　　　<0 モード取得
戻り値：d0.l=動作モード
　　　　関係ないビットは0になる。
備考：現時点では機能しない

ファンクション　$8005　　　M_SET_VOLTBL

機能：　MPCM.Xの音量マップの設定
引数：　d0.w=$8005
　　　　d1.l=音量テーブルセレクト
　　　　　　0: 128段階モード
　　　　　　1: 16段階モード(PCM8.X互換)
　　　　　　-1: ユーザー定義
　　　　a1.l=ユーザー定義の音量テーブルアドレス
　　　　　(d1.l=-1のときのみ有効)
戻り値：なし
備考：音量変換に使う音量マップを1/128の精度で設定する。音量マップは128個のワードデータから構成され、発音時の音量はこの値から決定される。
@v64に対応する定数を128とし、@v64が原音量で再生されるような設定にするのが望ましい。
なお、MPCMの内部処理の関係で、定数が
16,24,32,40,48,64,80,96,128
160,192,256,320,384,512,640
の場合、処理が高速化される。
参考として、d1.l=0(128段階) と d1.l=1(16段階)のテーブル内容を以下に示す。

・128段階モードの場合

	+0	+1	+2	+3	+4	+5	+6	+7
@v0	16,	17,	18,	19,	20,	21,	22,	23
@v8	24,	25,	26,	27,	28,	29,	30,	31
@v16	32,	33,	34,	35,	36,	37,	38,	39
@v24	40,	41,	42,	43,	44,	45,	46,	47
@v32	48,	50,	52,	54,	56,	58,	60,	62
@v40	64,	66,	68,	70,	72,	74,	76,	78
@v48	80,	82,	84,	86,	88,	90,	92,	94
@v56	96,	100,	104,	108,	112,	116,	120,	124
@v64	128,	132,	136,	140,	144,	148,	152,	156
@v72	160,	164,	168,	172,	176,	180,	184,	188
@v80	192,	200,	208,	216,	224,	232,	240,	248
@v88	256,	264,	272,	280,	288,	296,	304,	312
@v96	320,	328,	336,	344,	352,	360,	368,	376
@v104	384,	400,	416,	432,	448,	464,	480,	496
@v112	512,	528,	544,	560,	576,	592,	608,	624
@v120	640,	656,	672,	688,	704,	720,	736,	752

・16段階モードの場合

	+0	+1	+2	+3	+4	+5	+6	+7
@v0	16,	16,	16,	16,	16,	16,	16,	16
@v8	24,	24,	24,	24,	24,	24,	24,	24
@v16	32,	32,	32,	32,	32,	32,	32,	32
@v24	40,	40,	40,	40,	40,	40,	40,	40
@v32	48,	48,	48,	48,	48,	48,	48,	48
@v40	64,	64,	64,	64,	64,	64,	64,	64
@v48	80,	80,	80,	80,	80,	80,	80,	80
@v56	96,	96,	96,	96,	96,	96,	96,	96
@v64	128,	128,	128,	128,	128,	128,	128,	128
@v72	160,	160,	160,	160,	160,	160,	160,	160
@v80	192,	192,	192,	192,	192,	192,	192,	192
@v88	256,	256,	256,	256,	256,	256,	256,	256
@v96	320,	320,	320,	320,	320,	320,	320,	320
@v104	384,	384,	384,	384,	384,	384,	384,	384
@v112	512,	512,	512,	512,	512,	512,	512,	512
@v120	640,	640,	640,	640,	640,	640,	640,	640

ファンクション　$8006　　　M_SET_ECHCNT

機能：　MPCM.Xの効果音発声数を設定する。
引数：　d0.w=$8006
　　　　d1.l=効果音発声数(0-8)
戻り値：なし
備考：ファンクション$10xxで、チャンネル番号に$e0を指定した場合に、空きを探すチャンネルの範囲を設定する。空きを探す際に、チャンネル番号 0～d1.l-1 までを調べる。
ファンクション$10xxでチャンネル番号に$e0を使わずにチャンネル番号を直接指定した場合は、このファンクションの設定値は無視され、指定通りのチャンネルで演奏される。255を指定した場合も同様に無視される。

ファンクション　$8010 (Reserved)

ファンクション　$8011 (Reserved)

ファンクション　$8012 (Reserved)

ファンクション　$8013 (Reserved)

15.5. P8EMU.Xとは

P8EMU.XはMPCM.Xを使ってPCM8.Xのエミュレーションを行うソフトウェアで、Z-MUSIC ver.2.0など、PCM8.Xを前提に製作されたソフトなどをMPCM.Xを用いて利用する場合などに使用する。ただし、エミュレーションは完全ではないためPCM8.X対応のすべてのアプリケーションが使用可能になる、というわけではない。
使用する場合には、必ずMPCM.Xを常駐させたあとでP8EMU.Xを常駐させなければならない。

MEASURE 16

C言語ライブラリ

Z-MUSIC ver.3.0用のC言語ライブラリについて解説します。

16. 1. Z-MUSIC ver.3.0用C言語ライブラリの構成

Z-MUSIC ver.3.0用のC言語ライブラリは以下の4つのファイルで構成されている。

ZMSC3LIB.DEF　BC.X用DEFファイル
ZMSC3LIB.L　C言語用ライブラリ
ZMSC3LIB.A　C言語用ライブラリアーカイブファイル
ZMSC3LIB.H　C言語用インクルードファイル

16. 2. ZMSC3LIB. DEFの利用法

Z-MUSIC ver.3.0を利用したX-BASIC(MUSICZ3.FNC)のプログラムをコンパイルするときに使用する。まず、自分のシステムの、BC.X(X-BASICをC言語へコンバートするプログラム、C言語PRO68Kに付属)が格納してあるディレクトリにZMSC3LIB.DEFをコピーする。

例

```
A>COPY ZMSC3LIB.DEF A:¥BC
```

そのディレクトリ内のBASIC.CNFというファイルを開き、

```
FUNC=ZMSC3LIB
```

の1行を加えてセーブする。BASIC.CNF中に、

```
FUNC=MUSIC
FUNC=MUSIC2
FUNC=MUSIC3
FUNC=ZMUSIC
```

などが記述してある場合は、それらをすべて削除する(同名の関数が存在していると誤動作する可能性があるため)。

16. 3. ZMSC3LIB. Lの利用法

ライブラリファイルを利用するには、まず自分のシステムのライブラリファイルが格納してあるディレクトリにZMSC3LIB.Lをコピーする必要がある。

例

```
A>COPY ZMSC3LIB.L A:¥LIB
```

コンパイルしてできるオブジェクトにZMSC3LIB.Lをリンクし実行ファイルを得るには、

・**ZMSC3.Xを使用したX-BASIC(MUSICZ3.FNC)のプログラムをコンパイルする場合**

```
A>CC ????.BAS ZMSC3LIB.L
```

・**ZMSC3.Xを使用したC言語のプログラムをコンパイルする場合**

```
A>CC ????.C ZMSC3LIB.L
```

のようにする。

16. 4. ZMSC3LIB. Hの利用法

ZMSC3LIB.Hを利用するには、まず、自分のシステムの、インクルードファイルが格納してあるディレクトリに、これをコピーする必要がある。

例

```
A>COPY ZMSC3LIB.H A:¥INCLUDE
```

ZMSC3.XをC言語のプログラムから使いたい場合は、そのプログラムの冒頭に、

```
#INCLUDE    <ZMSC3LIB.H>
```

の1行をつける。

16. 5. ZMSC3LIB関数一覧

[注意]

備考欄の、

setup[7]:Output Assign　　(0～4:Dry,Rev,Cho,Dir1,Dir2)

といった表記は実際にsetup[7]に格納すべき値が0～4で、これに対応する音源上でのパラメータがDry,Rev,Cho,Dir1,Dir2ということを表す。

つまりこの例でいけば0がDryに、1がRev、2がChoに対応するということ。

int　zm_check_zmsc(void);

機能　Z-MUSICの常駐状態を検査する
引数　なし
戻り値　-1:エラー(常駐していなかった)
　　$3x4x:バージョンID
備考　バージョンIDはたとえばZMSC3.X ver.3.12が常駐していたならば$3142が返ってくる。4の部分は固定値。

int　zm_get_zmlibver(void);

機能　ライブラリのバージョンを返す
引数　なし
戻り値　$?.??:バージョンID
備考　バージョンIDはたとえば0.01ならば$001が返ってくる。

int　zm_init(int param);

機能　Z-MUSICと音源の初期化を行う
引数　int param:0(dummy)
戻り値　Z-MUSICのバージョンID
備考　paramは将来の拡張用で、現時点では必ず0を設定する。
　　バージョンIDはたとえばバージョン3.21ならば $00003241となる。
　　4の部分は固定値。

int　zm_assign(int device,int track);

機能　チャンネルとトラックの割り付けを設定する
引数　int device(上位16ビット):0=FM音源,1=AD PCM音源
　　$8000=MIDI1,$8001=MIDI2
　　$8002=MIDI3,$8003=MIDI4
　　-1=カレントMIDI
　int device(下位16ビット):チャンネル番号0-15
　int track:トラック番号0-65534
戻り値　0:正常終了
　　0以外:エラーコード
備考　カレントMIDIはMIDIインタフェイス番号を指定しない場合のデフォルトとなるものでzm_current_midi_out()で設定する

*int　zm_compiler(int mode,int size,char *zms,char **result);*

機能　ZMSをコンパイルする
引数　int mode:d0-d6:検出するエラーの最大値/0:∞,1-127
　　d15:エラーテーブルリストを作成して出力するか
　　(0:No,1:Yes)

int size:ZMSサイズ
char *zms:ZMSが格納されているアドレス
char **result:コンパイラが出力したZMD/エラーリストのアドレスを格納する変数

戻り値 0:正常終了(**resultにはZMDのアドレス)
0以外:コンパイルで発生したエラーの数
(**resultにはエラーテーブルのアドレス)

備考 エラーが発生するとchar **resultにエラーテーブルのアドレスが 代入される。char **resultはエラーリストを作成しない場合も必ず設定 すること。
エラーテーブルの構成についてはMEASURE11を参照のこと。
char **resultに返された領域は使用後zm_init()などを実行すればクリアされる。
char *zmsとint sizeを共に0にするとコンパイラの常駐解除となる。通常は指定してはならない。コンパイラは外部組み込みのファンクションであるため、使用前にはコンパイラの常駐を確認する必要がある。

*int zm_vget(int timbre, int mode, char *buffer);*

機能 FM音源音色を取り出す
引数 int timbre:音色番号(0-32767)
int mode:モード(0=OPMDRV.X互換,1=AL/FB分離方式)
char *buffer:音色データ格納バッファ
戻り値 0:正常終了
0以外:エラーコード
備考 char *bufferで表される音色データの書式は以下のとおり。

音色フォーマット(OPMDRV.X互換方式)

0:AF(ﾌｨｰﾄﾞﾊﾞｯｸ/ｱﾙｺﾞﾘｽﾞﾑ)	(0～63)
1:OM(ｽﾛｯﾄﾏｽｸ)	(0～15)
2:WF(ｳｪｰｳﾞﾌｫｰﾑ)	(0～3)
3:SYC(ｼﾝｸﾛ)	(0,1)
4:SPD(ｽﾋﾟｰﾄﾞ)	(0～255)
5:PMD	(0～127)
6:AMD	(0～127)
7:PMS	(0～7)
8:AMS	(0～3)
9:PAN	(0～3)
10:DUMMY	
11(OP1),22(OP2),33(OP3),44(OP4):AR(ｱﾀｯｸﾚｰﾄ)	(0～31)
12(OP1),23(OP2),34(OP3),45(OP4):1DR(1stﾃﾞｨｹｲﾚｰﾄ)	(0～31)
13(OP1),24(OP2),35(OP3),46(OP4):2DR(2ndﾃﾞｨｹｲﾚｰﾄ)	(0～31)
14(OP1),25(OP2),36(OP3),47(OP4):RR(ﾘﾘｰｽﾚｰﾄ)	(0～15)
15(OP1),26(OP2),37(OP3),48(OP4):1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)	(0～15)
16(OP1),27(OP2),38(OP3),49(OP4):TL(ﾄｰﾀﾙﾚﾍﾞﾙ)	(0～127)
17(OP1),28(OP2),39(OP3),50(OP4):KS(ｷｰｽｹｰﾘﾝｸﾞ)	(0～3)
18(OP1),29(OP2),40(OP3),51(OP4):MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)	(0～15)
19(OP1),30(OP2),41(OP3),52(OP4):DT1(ﾃﾞｨﾁｭｰﾝ1)	(0～7)
20(OP1),31(OP2),42(OP3),53(OP4):DT2(ﾃﾞｨﾁｭｰﾝ2)	(0～3)
21(OP1),32(OP2),43(OP3),54(OP4):AME(AMSｲﾈｰﾌﾞﾙ)	(0,1)

音色フォーマット(AL/FB分離方式)

00(OP1),11(OP2),22(OP3),33(OP4):AR(ｱﾀｯｸﾚｰﾄ)	(0～31)
01(OP1),12(OP2),23(OP3),34(OP4):1DR(1stﾃﾞｨｹｲﾚｰﾄ)	(0～31)
02(OP1),13(OP2),24(OP3),35(OP4):2DR(2ndﾃﾞｨｹｲﾚｰﾄ)	(0～31)
03(OP1),14(OP2),25(OP3),36(OP4):RR(ﾘﾘｰｽﾚｰﾄ)	(0～15)
04(OP1),15(OP2),26(OP3),37(OP4):1DL(1stﾃﾞｨｹｲﾚﾍﾞﾙ)	(0～15)
05(OP1),16(OP2),27(OP3),38(OP4):TL(ﾄｰﾀﾙﾚﾍﾞﾙ)	(0～127)
06(OP1),17(OP2),28(OP3),39(OP4):KS(ｷｰｽｹｰﾘﾝｸﾞ)	(0～3)
07(OP1),18(OP2),29(OP3),40(OP4):MUL(ﾌｪｰｽﾞﾏﾙﾁﾌﾟﾗｲ)	(0～15)
08(OP1),19(OP2),30(OP3),41(OP4):DT1(ﾃﾞｨﾁｭｰﾝ1)	(0～7)
09(OP1),20(OP2),31(OP3),42(OP4):DT2(ﾃﾞｨﾁｭｰﾝ2)	(0～3)
10(OP1),21(OP2),32(OP3),43(OP4):AME(AMSｲﾈｰﾌﾞﾙ)	(0,1)
44:AL(ｱﾙｺﾞﾘｽﾞﾑ)	(0～7)
45:FB(ﾌｨｰﾄﾞﾊﾞｯｸ)	(0～7)
46:OM(ｽﾛｯﾄﾏｽｸ)	(0～15)
47:PAN	(0～3)
48:WF(ｳｪｰｳﾞﾌｫｰﾑ)	(0～3)
49:SYC(ｼﾝｸﾛ)	(0,1)
50:SPD(ｽﾋﾟｰﾄﾞ)	(0～255)
51:PMD	(0～127)
52:AMD	(0～127)
53:PMS	(0～7)
54:AMS	(0～3)

*int zm_vset(int timbre, int mode, char *buffer);*

機能 FM音源音色を登録する
引数 int timbre:音色番号(0-32767)
int mode:モード(0=OPMDRV.X互換,1=AL/FB分離方式)
char *buffer:音色データ格納バッファ
戻り値 0:正常終了
0以外:エラーコード
備考 音色フォーマットについてはzm_vget()とまったく同じ

int zm_tempo(int tempo, int mode);

機能 テンポを設定する
引数 int tempo:テンポの値(1分間の4分音符の数),
-1テンポ取得のみを行う
int mode:0=Z-MUSICに登録する(通常)
-1=単なるテンポの変更でZ-MUSICには登録しない
戻り値 設定前のテンポとタイマ値が戻ってくる
上位16ビット:タイマ値
下位16ビット:テンポ値
備考 テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。
TIMER-A:20～32767
TIMER-B:20～5000
YM3802:1～32767

int zm_set_timer_value(int timer, int mode);

機能 テンポをタイマ値で設定する
引数 int timer:タイマの値,
-1=タイマ値取得のみを行う
int mode:0=Z-MUSICに登録する(通常)
-1=単なるテンポの変更でZ-MUSICには登録しない
戻り値 設定前のテンポとタイマ値が戻ってくる
上位16ビット:テンポ値
下位16ビット:タイマ値
備考 テンポ値の有効範囲は使用するテンポ源タイマの種類によって異なる。
TIMER-A:0～1023
TIMER-B:0～255
YM3802:0～8191

*short *zm_play(short *track);*

機能 演奏開始制御
引数 short *track:演奏を開始したいトラック番号の並び
戻り値 演奏開始を行えなかったトラック番号の並び
備考 short *track,戻り値共に、
{ short track,short track,.....,short track,(short)-1}
という構成を取る。ただしトラック番号は0-65534の範囲

void zm_play_all(void);

機能 全トラック演奏開始制御
引数 なし
戻り値 なし

void zm_play2(void);

機能 演奏開始制御
引数 なし
戻り値 なし
備考 現在Z-MUSICの演奏データバッファに存在する演奏データをはじめから(共通コマンドの実行も含めて)演奏する。

void zm_play_again(void);

機能 演奏開始制御
引数 なし
戻り値 なし

備考　前回行われたzm_play()を再び実行する

*void zm_play_status_all_ch(int *channel_list);*

機能　全チャンネル演奏状態検査
引数　int *channel_list:検査結果を格納するためのバッファ
戻り値　なし
備考　演奏状態であるチャンネルIDが*channel_listに格納される。
{(int)active_device,(int)active_device,...,(int)-1 }
のような構成を取る。
バッファサイズは演奏している(トラック数+1)*4バイト必要。
なおチャンネルIDは、
(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI
(下位16ビット):チャンネル番号0-15
で表される。

*void zm_play_status_all_tr(short *track_list);*

機能　全トラック演奏状態検査
引数　short *track_list:検査結果を格納するためのバッファ
戻り値　なし
備考　演奏状態であるトラック番号(0-65534の範囲)が*track_listに格納される。
{(short)active_track,(short)active_track,...,(short)-1}
のような構成を取る。
バッファサイズは演奏している(トラック数+1)*2バイト必要。

int zm_play_status_ch(int channel);

機能　チャンネル演奏状態検査
引数　int channel(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI
int channel(下位16ビット):チャンネル番号0-15
戻り値　0:演奏されていない
0以外:演奏されている

int zm_play_status_tr(int track);

機能　トラック演奏状態検査
引数　int track:検査対象トラック番号(0-65534)
戻り値　0:演奏されていない
0以外:演奏されている

*short *zm_stop(short *track);*

機能　演奏停止制御
引数　short *track:演奏を停止したいトラック番号の並び
戻り値　演奏停止を行えなかったトラック番号の並び
備考　short *track,戻り値共に、
{ short track,short track,.....,short track,(short)-1}
という構成を取る。ただしトラック番号は0-65534の範囲

void zm_stop_all(void);

機能　全トラック演奏停止制御
引数　なし
戻り値　なし

*short *zm_cont(short *track);*

機能　演奏再開制御
引数　short *track:演奏を再開したいトラック番号の並び
戻り値　演奏再開を行えなかったトラック番号の並び
備考　short *track,戻り値共に、
{ short track,short track,.....,short track,(short)-1}
という構成を取る。ただしトラック番号は0-65534の範囲

void zm_cont_all(void);

機能　全トラック演奏再開制御
引数　なし
戻り値　なし

*char *zm_atoi(int track);*

機能　演奏トラックデータの先頭アドレス取得
引数　int track:トラック番号0-65534
戻り値　0:エラー
0以外:指定演奏トラックの先頭アドレス

*MSTRCLK *zm_set_master_clock(int side, MSTRCLK *mstrclk);*

機能　拍子,メトロノーム速度,調号,全音符の絶対音長の設定
引数　int side:0=音楽演奏,1=効果音
MSTRCLK mstrclk:拍子,メトロノーム速度,調号,全音符の絶対音長を格納した構造体のアドレス。
0ならば問い合わせのみ
戻り値　MSTRCLK mstrclk=0としたときのみ、その時点の設定値を格納したMSTRCLK構造体のアドレスが返る。それ以外は戻り値不定。
備考　side=0で通常音楽演奏側の設定、side=1で効果音演奏側の設定となる。
MSTRCLK構造体

```
struct  MSTRCLK {
  char   meter_n;/*拍子分子(1-255)*/
  char   meter_d;/*拍子分母(1-255)*/
  char   metronome;       /*音楽的音長(1-255)*/
  char   dummy;
  char   key1;            /*調号1(0:C,負値:♭の数,
                          /*          正値:#の数)*/
  char   key2;            /*調号2(Major=0,Minor=1)*/
  short  master_clock;    /*全音符の絶対音長(1-65535)*/
};
```

*int zm_play_zmd(int size, char *zmd);*

機能　ZMDの演奏
引数　int size:ZMDのサイズ
(ヘッダ$1A,'ZmuSiC',$30の分は含まれていても含まれていなくてもどちらでもよい)
０を設定するとZ-MUSIC本体のバッファへ演奏データを取り込む動作を省略する。
char *zmd:ZMDの格納バッファ(ヘッダを含まず)
戻り値　0:正常終了
0以外:エラーコード
備考　int size=0とするには、char *zmdで表されるアドレスからの領域の存在が演奏中も保証されることを前提とする。

*int zm_play_zmd_se(char *zmd);*

機能　ZMDを効果音として演奏する
引数　char *zmd:ZMDの格納バッファ(ヘッダを含まず)
戻り値　０:正常終了
０以外:エラーコード
備考　char *zmdで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。

*int zm_se_play(char *zmd);*

機能　効果音ZMDの演奏
引数　char *zmd:ZMDの格納バッファ(トラック・チャンネル情報テーブルから)
戻り値　0:正常終了
0以外:エラーコード
備考　char *zmdで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。トラック・チャンネル情報テーブルについてはMEASURE12を参照。

*int zm_se_adpcm1(char data_type, char volume, char frq, char pan, int size, short priority, short ch, char *adpcm);*

機能　(AD)PCM効果音の演奏
引数　char data_type:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)
char volume:音量(0-127,64:標準)
char frq:再生周波数
(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz)

char pan:パンポット(0-3 / $80+0-127)
int size:再生するAD PCMのデータサイズ
short priority:効果音優先度(0-255/0がもっとも低い)
short ch:再生チャンネル(0-15)
char *adpcm:(AD)PCMデータアドレス

戻り値 0:正常終了
0以外:エラーコード

備考 char *adpcmで表されるアドレスからの領域の存在は、演奏中も保証されなければならない。

int zm_se_adpcm2(char data_type, char volume, char frq, char pan, int data_number, short priority, short ch);

機能 Z-MUSICに登録してある(AD)PCM効果音の演奏

引数 char data_type:データタイプ(-1:AD PCM,1:16bitPCM,2:8bitPCM)
char volume:音量(0-127,64:標準)
char frq:再生周波数
(0:3.9kHz 1:5.2kHz 2:7.8kHz 3:10.4kHz 4:15.6kHz)
char pan:パンポット(0-3 / $80+0-127)
int data_number:PCMデータ番号
(d15が1ならば音色番号($8000+0-32767),
0ならばノート番号(0-32767)指定)
short priority:効果音優先度(0-255/0がもっとも低い)
short ch:再生チャンネル(0-15)

戻り値 0:正常終了
0以外:エラーコード

void zm_intercept_play(int mode);

機能 演奏開始制御の遮断制御

引数 int mode:-1 演奏開始の遮断を行う
0 演奏開始の遮断の解除と演奏開始
1 演奏開始の遮断の解除

戻り値 なし

備考 演奏開始の遮断を設定するとzm_play()などの演奏開始命令を実行しても演奏が開始されなくなる。
演奏開始の遮断中に演奏開始を実行した場合、ワークtimer_flgの最上位ビットが1に上書きされる。

int zm_current_midi_in(int port);

機能 カレントMIDI-IN端子の設定

引数 int port:MIDIポートの番号(0-3)
-1では問い合わせのみを行う

戻り値 以前のカレントMIDI-INのポート番号(0-3)

int zm_current_midi_out (int port);

機能 カレントMIDI-OUT端子の設定

引数 int port:MIDIポートの番号(0-3)
-1では問い合わせのみを行う

戻り値 以前のカレントMIDI-OUTのポート番号(0-3)

*int zm_midi_transmission(int port, int size, char *midi);*

機能 MIDIデータメッセージの送信

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:送信データサイズ
0とした場合はテキスト型MIDIデータ(MDD)とみなして送信する
0以外ではバイナリデータとみなして送信する
char *midi:送信するMIDIデータを格納したデータバッファ

戻り値 送信データの8ビット加算合計値

備考 MDDとはZ-MUSIC ver.2.0以前でMIDIダンプファイルの標準としていたフォーマット。送信データサイズを指定した場合はバイナリのMIDIメッセージとみなして送信される。Z-MUSICが扱えるバイナリのMIDIメッセージにはスタンダードMIDIファイル(FORMAT0)と、単なるバイナリデータ列の２種類がありこのデータの種類判別はZ-MUSIC ver.3.0が自動で行う。

*void zm_exclusive(int port, int size, int id, char *midi);*

機能 エクスクルーシブメッセージの送信

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:送信データサイズ
int id:d0-d7=MODEL ID(0-127)
d8-d15=DEVICE ID(0-127)
d16-d23=MAKER ID(0-127)
char *midi:送信するMIDIデータを格納したデータバッファ

戻り値 なし

備考 エクスクルーシブヘッダの$F0とEOXの$F7は自動送信される。
DEVICE IDを-1とすると以前設定したものが選択される。
最初の使用時に省略した場合はドライバ内のデフォルト値が選択される。
MAKER IDをRoland($41)とした場合はデータ列をローランドエクスクルーシブフォーマットのデータとみなし、コマンドメッセージ(DT2:$12)やチェックサムバイト付きで送信する。
MAKER IDをYAMAHA($43)とした場合はデータ列をヤマハエクスクルーシブフォーマットのデータとみなし、データ長やチェックサムバイト付きで送信する。

int zm_set_eox_wait (int port, int wait);

機能 EOX送信後のウェイトを設定する

引数 int port:MIDIポートの番号(0-2)
-1はカレントMIDI-OUTを使用
int wait:0-65535

戻り値 それまでのウェイト値

int zm_midi_inp1 (int port, int mode);

機能 MIDIデータの1バイト入力

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int mode:0 1回受信を試みてすぐ帰還
1 受信するまで無限ループ

戻り値 d31=1 読みこぼし発生 0:正常読み出し
d15=1 データなし 0:データあり
d0-d7 読み出しデータ

int zm_midi_out1 (int port, char data);

機能 MIDIデータの1バイト出力

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
char data:送信データ(0-255)

戻り値 0:正常終了
0以外:エラーコード

int zm_midi_rec (int port);

機能 MIDIデータのレコーディング開始

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用

戻り値 0:正常終了
0以外:エラーコード

int zm_midi_rec_end (int port);

機能 MIDIデータのレコーディング終了

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用

戻り値 0:正常終了
0以外:エラーコード

int zm_gs_reset (int port, int id);

機能 GS音源の初期化

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:DEVICE ID(0-127,-1:以前の値を使用)

戻り値 0:正常終了
0以外:エラーコード

*int zm_gs_partial_reserve(int port, int id, char *rsv);*

機能　GS音源のパーシャルリザーブ

引数　int port:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *rsv:パート1～パート16までのパーシャルリザーブ値が
　　格納されているアドレス

戻り値　0:正常終了
0以外:エラーコード

備考　char *rsvからの領域にパーシャルリザーブ値を16パート分格納しておく。また、合計が楽器の最大同時発声数以下にならなければならない。
rsv[0]＝パート1,rsv[1]＝パート2…,rsv[15]＝パート16
というふうに対応する。なお、パート10はリズムパートに対応する。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

*int zm_gs_reverb(int port, int size, int id, char *rvb);*

機能　GS音源のリバーブパラメータの設定

引数　int port:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
int size:リバーブパラメータ・データサイズ(1-7)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *rvb:リバーブパラメータ格納バッファ

戻り値　0:正常終了
0以外:エラーコード

備考　char *rvbからの領域にリバーブパラメータを記述しておく。
sizeは7以内で、指定したsize個分のパラメータがGS音源へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

```
rvb[0]:REVERB MACRO      (00:Room1,01:Room2,02:Room3,
                          03:Hall1,04:Hall2,05:Plate,
                          06:Delay,07:Panning Delay(FB))
rvb[1]:REVERB CHARACTER              ($00～$07)
rvb[2]:REVERB PRE-LPF                ($00～$07)
rvb[3]:REVERB LEVEL                  ($00～$7F)
rvb[4]:REVERB TIME                   ($00～$7F)
rvb[5]:REVERB DELAY FEEDBACK         ($00～$7F)
rvb[6]:REVERB SEND LEVEL TO CHORUS   ($00～$7F)
```

(SC-55マニュアルp79参照)

*int zm_gs_chorus(int port, int size, int id, char *cho);*

機能　GS音源のコーラスパラメータの設定

引数　int port:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
int size:コーラスパラメータデータサイズ(1-8)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *cho:コーラスパラメータ格納バッファ

戻り値　0:正常終了
0以外:エラーコード

備考　char *choからの領域にコーラスパラメータを記述しておく。
sizeは8以内で指定したsize個分のパラメータがGS音源へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
cho[0]:CHORUS MACRO      (00:Chorus1,01:Chorus2,02:Chorus3,
                          03:Chorus4,04:Feedback Chorus,
                          05:Flanger,06:Short Delay,
                          07:Short Delay(FB))
cho[1]:CHORUS PRE-LPF                ($00～$07)
cho[2]:CHORUS LEVEL                  ($00～$7F)
cho[3]:CHORUS FEEDBACK               ($00～$7F)
cho[4]:CHORUS DELAY                  ($00～$7F)
cho[5]:CHORUS RATE                   ($00～$7F)
cho[6]:CHORUS DEPTH                  ($00～$7F)
cho[7]:CHORUS SEND LEVEL TO REVERB   ($00～$7F)
```

(SC-55マニュアルp79参照)

*int zm_gs_part_parameter(int port, int size, int pt_id, char *param);*

機能　GS音源のパートパラメータの設定

引数　int port:MIDIポートの番号(0-3)
　　-1はカレントMIDI-OUTを使用
int size:パートパラメータデータサイズ(1-119)
int pt_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
　　d16-d31=PART NUMBER(1-16)
char *param:パートパラメータ格納バッファ

戻り値　0:正常終了
0以外:エラーコード

備考　char *paramからの領域にパートパラメータを記述しておく。
sizeは119以内で指定したsize個分のパラメータがGS音源へ送信される。
pt_idのd0-d15を-1とすると以前設定したものが選択される。
最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
param[0]:Rx CHANNEL               (1～16,17＝OFF)
param[1]:Rx PITCH BEND            (0～1:OFF/ON)
param[2]:Rx CH PRESSURE(CAf)      (0～1:OFF/ON)
param[3]:Rx PROGRAM CHANGE        (0～1:OFF/ON)
param[4]:Rx CONTROL CHANGE        (0～1:OFF/ON)
param[5]:Rx POLY PRESSURE(PAf)    (0～1:OFF/ON)
param[6]:Rx NOTE MESSAGE          (0～1:OFF/ON)
param[7]:Rx RPN                   (0～1:OFF/ON)
param[8]:Rx NRPN                  (0～1:OFF/ON)
param[9]:Rx MODURATION            (0～1:OFF/ON)
param[10]:Rx VOLUME               (0～1:OFF/ON)
param[11]:Rx PANPOT               (0～1:OFF/ON)
param[12]:Rx EXPRESSION           (0～1:OFF/ON)
param[13]:Rx HOLD1                (0～1:OFF/ON)
param[14]:Rx PORTAMENTO           (0～1:OFF/ON)
param[15]:Rx SOSTENUTO            (0～1:OFF/ON)
param[16]:Rx SOFT                 (0～1:OFF/ON)
param[17]:MONO/POLY MODE          (0～1:MONO/POLY)
param[18]:ASSIGN MODE
                 (0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
param[19]:USE FOR RHYTHM PART     (0:OFF,1:MAP1,2:MAP2)
param[20]:PITCH KEY SHIFT   ($28～$58:－24～＋24[semitone])
param[21]:PITCH OFFSET FINE(upper)
                            ($08～$F8:－12.0～＋12.0[Hz])
param[22]:PITCH OFFSET FINE(lower)        〃
param[23]:PART LEVEL              (0～127)
param[24]:VELOCITY SENSE DEPTH    (0～127)
param[25]:VELOCITY SENSE OFFSET   (0～127)
param[26]:PART PANPOT   (0:RANDOM,01(LEFT)～64～127(RIGHT))
param[27]:KEY RANGE LOW           (0～127:C-1～G9)
param[28]:KEY RANGE HIGH          (0～127:C-1～G9)
param[29]:CC1 CONTROL NUMBER      (0～127)
param[30]:CC2 CONTROL NUMBER      (0～127)
param[31]:CHORUS SEND DEPTH       (0～127)
param[32]:REVERB SEND DEPTH       (0～127)

param[33]:Vibrato rate            ($0E～$72:－50～＋50)
param[34]:Vibrato depth           ($0E～$72:－50～＋50)
param[35]:TVF cutoff freq.        ($0E～$50:－50～＋16)
param[36]:TVF resonance           ($0E～$72:－50～＋50)
param[37]:TVF&TVA Env.attack      ($0E～$72:－50～＋50)
param[38]:TVF&TVA Env.decay       ($0E～$72:－50～＋50)
param[39]:TVF&TVA Env.release     ($0E～$72:－50～＋50)
param[40]:Vibrato delay           ($0E～$72:－50～＋50)

param[41]:SCALE TUNING C      ($00～$7F:－64～＋63[cent])
param[42]:SCALE TUNING C#     ($00～$7F:－64～＋63[cent])
param[43]:SCALE TUNING D      ($00～$7F:－64～＋63[cent])
param[44]:SCALE TUNING D#     ($00～$7F:－64～＋63[cent])
param[45]:SCALE TUNING E      ($00～$7F:－64～＋63[cent])
param[46]:SCALE TUNING F      ($00～$7F:－64～＋63[cent])
param[47]:SCALE TUNING F#     ($00～$7F:－64～＋63[cent])
```

param[48]:SCALE TUNING G ($00～$7F:－64～＋63[cent])
param[49]:SCALE TUNING G# ($00～$7F:－64～＋63[cent])
param[50]:SCALE TUNING A ($00～$7F:－64～＋63[cent])
param[51]:SCALE TUNING A# ($00～$7F:－64～＋63[cent])
param[52]:SCALE TUNING B ($00～$7F:－64～＋63[cent])

param[53]:MOD PITCH CONTROL ($28～$58:－24～＋24[semitone])
param[54]:MOD TVF CUTOFF CONTROL
($00～$7F:-9600～+9600[CENT])
param[55]:MOD AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[56]:MOD LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[57]:MOD LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[58]:MOD LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[59]:MOD LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[60]:MOD LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[61]:MOD LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[62]:MOD LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[63]:MOD LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param[64]:BEND PITCH CONTROL($28～$58:－24～＋24[semitone])
param[65]:BEND TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param[66]:BEND AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[67]:BEND LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[68]:BEND LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[69]:BEND LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[70]:BEND LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[71]:BEND LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[72]:BEND LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[73]:BEND LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[74]:BEND LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param[75]:CAf PITCH CONTROL ($28～$58:－24～＋24[semitone])
param[76]:CAf TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param[77]:CAf AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[78]:CAf LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[79]:CAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[80]:CAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[81]:CAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[82]:CAf LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[83]:CAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[84]:CAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[85]:CAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param[86]:PAf PITCH CONTROL ($28～$58:－24～＋24[semitone])
param[87]:PAf TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param[88]:PAf AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[89]:PAf LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[90]:PAf LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[91]:PAf LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[92]:PAf LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[93]:PAf LFO2 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[94]:PAf LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[95]:PAf LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[96]:PAf LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param[97]:CC1 PITCH CONTROL ($28～$58:－24～＋24[semitone])
param[98]:CC1 TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param[99]:CC1 AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[100]:CC1 LFO1 RATE CONTROL
($00～$7F:－10.0～＋10.0[Hz])
param[101]:CC1 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[102]:CC1 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[103]:CC1 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[104]:CC1 LFO2 RATE CONTROL($00～$7F:－10.0～＋10.0[Hz])
param[105]:CC1 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[106]:CC1 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[107]:CC1 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])

param[108]:CC2 PITCH CONTROL ($28～$58:－24～＋24[semitone])
param[109]:CC2 TVF CUTOFF CONTROL
($00～$7F:－9600～＋9600[CENT])
param[110]:CC2 AMPLITUDE CONTROL
($00～$7F:－100.0～＋100.0[%])
param[111]:CC2 LFO1 RATE CONTROL($00～$7F:－10.0～＋10.0[Hz])
param[112]:CC2 LFO1 PITCH DEPTH ($00～$7F:0～600[cent])
param[113]:CC2 LFO1 TVF DEPTH ($00～$7F:0～2400[cent])
param[114]:CC2 LFO1 TVA DEPTH ($00～$7F:0～100.0[%])
param[115]:CC2 LFO2 RATE CONTROL($00～$7F:－10.0～＋10.0[Hz])
param[116]:CC2 LFO2 PITCH DEPTH ($00～$7F:0～600[cent])
param[117]:CC2 LFO2 TVF DEPTH ($00～$7F:0～2400[cent])
param[118]:CC2 LFO2 TVA DEPTH ($00～$7F:0～100.0[%])
(SC-55のマニュアルp75,p79～p80参照)

*int zm_gs_drum_setup(int port, int size, int dr_id, char *param);*

機能 GS音源のドラムセットアップ設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1-8)
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d23=NOTE NUMBER(0-127)
d24-d31=MAP NUMBER(0-1)
char *param:ドラムセットアップパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *paramからの領域にドラムセットアップパラメータを記述しておく。sizeは8以内で指定したsize個分のパラメータがGS音源へ送信される。idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

param[0]:PLAY KEY NUMBER (0～127)
param[1]:LEVEL (0～127)
param[2]:ASSIGN GROUP NUMBER (0:NON,1～127)
param[3]:PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
param[4]:REVERB DEPTH (0～127:0.0～1.0)
param[5]:CHORUS DEPTH (0～127:0.0～1.0)
param[6]:Rx NOTE OFF (0～1:OFF/ON)
param[7]:Rx NOTE ON (0～1:OFF/ON)
(SC-55マニュアルp75,p82参照)

*int zm_gs_drum_name(int port, int size, int mp_id, char *name);*

機能 GS音源のドラムセット名設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1-8)
int mp_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=MAP NUMBER(0,1)
char *name:ドラムセット名文字列格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 文字列は12文字以内でchar *nameからの領域に文字列を記述しておく。
idを-1とすると以前設定したものが選択される。最初の使用時に

省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
(SC-55マニュアルp82参照)

int zm_gs_print (int port, int size, int id, char *mes);

機能 GS音源の画面へのメッセージ出力

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:メッセージ長(1-32)
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *mes:文字列格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 メッセージは32文字以内でchar *mesからの領域に文字列を記述しておく。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

int zm_gs_display (int port, int id, short *pattern);

機能 GS音源の画面へのドットパターン出力

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
short *pattern:ドットパターン格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 short *patternからの領域にドットパターンを記述しておく。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

例

```
/*SC-55の画面に"善"を表示する*/
short d[16]={   %0001000000010000,
                %0000100000100000,
                %0111111111111100,
                %0000000100000000,
                %0011111111111000,
                %0000000100000000,
                %0111111111111100,
                %0001000100010000,
                %0000100100100000,
                %1111111111111110,
                %0000000000000000,
                %0011111111111000,
                %0010000000001000,
                %0010000000001000,
                %0011111111111000,
                %0010000000001000}
gs_display(-1,-1,d);
```

int zm_gm_system_on (int port);

機能 GM音源のリセット

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用

戻り値 0:正常終了
0以外:エラーコード

int zm_sc88_mode_set (int port, int md_id);

機能 SC-88モード設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int md_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=SC-88 MODE(0:Single, 1:Double)

戻り値 0:正常終了
0以外:エラーコード

備考 idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

int zm_sc88_reverb(int port, int size, int id, char *rvb);

機能 SC-88のリバーブパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:リバーブパラメータデータサイズ(1-8)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *rvb:リバーブパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *rvbからの領域にリバーブパラメータを記述しておく。
sizeは8以内で、指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

```
rvb[0]:REVERB MACRO      (00:Room1,01:Room2,02:Room3,
                          03:Hall1,04:Hall2,05:Plate,
                          06:Delay,07:Panning Delay(FB))
rvb[1]:REVERB CHARACTER               ($00～$07)
rvb[2]:REVERB PRE-LPF                 ($00～$07)
rvb[3]:REVERB LEVEL                   ($00～$7F)
rvb[4]:REVERB TIME                    ($00～$7F)
rvb[5]:REVERB DELAY FEEDBACK          ($00～$7F)
rvb[6]:REVERB SEND LEVEL TO CHORUS($00～$7F/SC-88ではﾀﾞﾐｰ)
rvb[7]:REVERB PRE-DELAY TIME          ($00～$7F)
```

(SC-88VLマニュアルp77参照)

int zm_sc88_chorus(int port, int size, int id, char *cho);

機能 SC-88のコーラスパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:コーラスパラメータ・データサイズ(1-9)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *cho:コーラスパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *choからの領域にコーラスパラメータを記述しておく。
sizeは9以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
cho[0]:CHORUS MACRO      (00:Chorus1,01:Chorus2,02:Chorus3,
                          03:Chorus4,04:Feedback Chorus,
                          05:Flanger,06:Short Delay,
                          07:Short Delay(FB))
cho[1]:CHORUS PRE-LPF                 ($00～$07)
cho[2]:CHORUS LEVEL                   ($00～$7F)
cho[3]:CHORUS FEEDBACK                ($00～$7F)
cho[4]:CHORUS DELAY                   ($00～$7F)
cho[5]:CHORUS RATE                    ($00～$7F)
cho[6]:CHORUS DEPTH                   ($00～$7F)
cho[7]:CHORUS SEND LEVEL TO REVERB    ($00～$7F)
cho[8]:CHORUS SEND LEVEL TO DELAY     ($00～$7F)
```

(SC-88VLマニュアルp78参照)

int zm_sc88_delay(int port, int size, int id, char *dly);

機能 SC-88のディレイパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ディレイパラメータデータサイズ(1-11)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *dly:ディレイパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *dlyからの領域にリバーブパラメータを記述しておく。
sizeは11以内で、指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される

```
dly[0]:DELAY MACRO       (00:Delay1,01:Delay2,02:Delay3,
                          03:Delay4,04:Pan Delay1,
                          05:Pan Delay2,06:Pan Delay3,
```

07:Pan Delay4,08:Delay to Reverb,
09:Pan Repeat)

dly[1]:DELAY PRE-LPF ($00〜$07)
dly[2]:DELAY TIME CENTER ($01〜$73:0.1ms〜sec)
dly[3]:DELAY TIME RATIO LEFT ($01〜$78:4〜500%)
dly[4]:DELAY TIME RATIO RIGHT ($01〜$78:4〜500%)
dly[5]:DELAY LEVEL CENTER ($00〜$7F)
dly[6]:DELAY LEVEL LEFT ($00〜$7F)
dly[7]:DELAY LEVEL RIGHT ($00〜$7F)
dly[8]:DELAY LEVEL ($00〜$7F)
dly[9]:DELAY FEEDBACK ($00〜$7F:-64〜+63)
dly[10]:DELAY SENDLEVEL TO REVERB ($00〜$7F)
(SC-88VLマニュアルp78参照)

*int zm_sc88_equalizer(int port, int size, int id, char *eql);*

機能 SC-88のイコライザパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:イコライザパラメータデータサイズ(1-4)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *eql:イコライザパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *eqlからの領域にコーラスパラメータを記述しておく。
sizeは4以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

eql[0]:EQ LOW FREQ (0,1:200Hz,400Hz)
eql[1]:EQ LOW GAIN ($34〜$4c:-12〜+12dB)
eql[2]:EQ HIGH FREQ (0,1:3kHz,6kHz)
eql[3]:EQ HIGH GAIN ($34〜$4c:-12〜+12dB)
(SC-88VLマニュアルp78参照)

*int zm_sc88_part_parameter(int port, int size, int pt_id, char *param);*

機能 SC-88のパートパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:パートパラメータデータサイズ(1-127)
int pt_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=PART NUMBER(1-16)
char *param:パートパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *paramからの領域にパートパラメータを記述しておく。
sizeは127以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

param[0]:Rx CHANNEL (1〜16,17=OFF)
param[1]:Rx PITCH BEND (0〜1:OFF/ON)
param[2]:Rx CH PRESSURE(CAf) (0〜1:OFF/ON)
param[3]:Rx PROGRAM CHANGE (0〜1:OFF/ON)
param[4]:Rx CONTROL CHANGE (0〜1:OFF/ON)
param[5]:Rx POLY PRESSURE(PAf) (0〜1:OFF/ON)
param[6]:Rx NOTE MESSAGE (0〜1:OFF/ON)
param[7]:Rx RPN (0〜1:OFF/ON)
param[8]:Rx NRPN (0〜1:OFF/ON)
param[9]:Rx MODURATION (0〜1:OFF/ON)
param[10]:Rx VOLUME (0〜1:OFF/ON)
param[11]:Rx PANPOT (0〜1:OFF/ON)
param[12]:Rx EXPRESSION (0〜1:OFF/ON)
param[13]:Rx HOLD1 (0〜1:OFF/ON)
param[14]:Rx PORTAMENTO (0〜1:OFF/ON)
param[15]:Rx SOSTENUTO (0〜1:OFF/ON)
param[16]:Rx SOFT (0〜1:OFF/ON)
param[17]:MONO/POLY MODE (0〜1:MONO/POLY)
param[18]:ASSIGN MODE
(0:SINGLE,1:LIMITED-MULTI,2:FULL-MULTI)
param[19]:USE FOR RHYTHM PART (0:OFF,1:MAP1,2:MAP2)
param[20]:PITCH KEY SHIFT ($28〜$58:−24〜+24[semitone])
param[21]:PITCH OFFSET FINE(upper)
($08〜$F8:−12.0〜+12.0[Hz])
param[22]:PITCH OFFSET FINE(lower) 〃
param[23]:PART LEVEL (0〜127)
param[24]:VELOCITY SENSE DEPTH (0〜127)
param[25]:VELOCITY SENSE OFFSET (0〜127)
param[26]:PART PANPOT (0:RANDOM,01(LEFT)〜64〜127(RIGHT))
param[27]:KEY RANGE LOW (0〜127:C-1〜G9)
param[28]:KEY RANGE HIGH (0〜127:C-1〜G9)
param[29]:CC1 CONTROL NUMBER (0〜127)
param[30]:CC2 CONTROL NUMBER (0〜127)
param[31]:CHORUS SEND LEVEL (0〜127)
param[32]:REVERB SEND LEVEL (0〜127)
param[33]:Rx BANK SELECT (0:OFF,1:ON)
param[34]:Rx BANK SELECT LSB (0:OFF,1:ON)

param[35]:PITCH FINE TUNE(Upper)
($00 00〜$40 00〜$7f 7f:-100〜0〜+100)
param[36]:PITCH FINE TUNE(Lower)

param[37]:DELAY SEND LEVEL (0〜127)

param[38]:Vibrato rate ($00〜$7f:−64〜+63)
param[39]:Vibrato depth ($00〜$7f:−64〜+63)
param[40]:TVF cutoff freq. ($00〜$7f:−64〜+63)
param[41]:TVF resonance ($00〜$7f:−64〜+63)
param[42]:TVF&TVA Env.attack ($00〜$7f:−64〜+63)
param[43]:TVF&TVA Env.decay ($00〜$7f:−64〜+63)
param[44]:TVF&TVA Env.release ($00〜$7f:−64〜+63)
param[45]:Vibrato delay ($00〜$7f:−64〜+63)

param[46]:SCALE TUNING C ($00〜$7F:−64〜+63[cent])
param[47]:SCALE TUNING C# ($00〜$7F:−64〜+63[cent])
param[48]:SCALE TUNING D ($00〜$7F:−64〜+63[cent])
param[49]:SCALE TUNING D# ($00〜$7F:−64〜+63[cent])
param[50]:SCALE TUNING E ($00〜$7F:−64〜+63[cent])
param[51]:SCALE TUNING F ($00〜$7F:−64〜+63[cent])
param[52]:SCALE TUNING F# ($00〜$7F:−64〜+63[cent])
param[53]:SCALE TUNING G ($00〜$7F:−64〜+63[cent])
param[54]:SCALE TUNING G# ($00〜$7F:−64〜+63[cent])
param[55]:SCALE TUNING A ($00〜$7F:−64〜+63[cent])
param[56]:SCALE TUNING A# ($00〜$7F:−64〜+63[cent])
param[57]:SCALE TUNING B ($00〜$7F:−64〜+63[cent])

param[58]:MOD PITCH CONTROL ($28〜$58:−24〜+24[semitone])
param[59]:MOD TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param[60]:MOD AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param[61]:MOD LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param[62]:MOD LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])
param[63]:MOD LFO1 TVF DEPTH ($00〜$7F:0〜2400[cent])
param[64]:MOD LFO1 TVA DEPTH ($00〜$7F:0〜100.0[%])
param[65]:MOD LFO2 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param[66]:MOD LFO2 PITCH DEPTH ($00〜$7F:0〜600[cent])
param[67]:MOD LFO2 TVF DEPTH ($00〜$7F:0〜2400[cent])
param[68]:MOD LFO2 TVA DEPTH ($00〜$7F:0〜100.0[%])

param[69]:BEND PITCH CONTROL($40〜$58:−24〜+24[semitone])
param[70]:BEND TVF CUTOFF CONTROL
($00〜$7F:−9600〜+9600[CENT])
param[71]:BEND AMPLITUDE CONTROL
($00〜$7F:−100.0〜+100.0[%])
param[72]:BEND LFO1 RATE CONTROL
($00〜$7F:−10.0〜+10.0[Hz])
param[73]:BEND LFO1 PITCH DEPTH ($00〜$7F:0〜600[cent])

param[74]:BEND LFO1 TVF DEPTH ($00~$7F:0~2400[cent])
param[75]:BEND LFO1 TVA DEPTH ($00~$7F:0~100.0[%])
param[76]:BEND LFO2 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[77]:BEND LFO2 PITCH DEPTH ($00~$7F:0~600[cent])
param[78]:BEND LFO2 TVF DEPTH ($00~$7F:0~2400[cent])
param[79]:BEND LFO2 TVA DEPTH ($00~$7F:0~100.0[%])

param[80]:CAf PITCH CONTROL ($28~$58:－24~＋24[semitone])
param[81]:CAf TVF CUTOFF CONTROL
($00~$7F:－9600~＋9600[CENT])
param[82]:CAf AMPLITUDE CONTROL
($00~$7F:－100.0~＋100.0[%])
param[83]:CAf LFO1 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[84]:CAf LFO1 PITCH DEPTH ($00~$7F:0~600[cent])
param[85]:CAf LFO1 TVF DEPTH ($00~$7F:0~2400[cent])
param[86]:CAf LFO1 TVA DEPTH ($00~$7F:0~100.0[%])
param[87]:CAf LFO2 RATE CONTROL($00~$7F:－10.0~＋10.0[Hz])
param[88]:CAf LFO2 PITCH DEPTH ($00~$7F:0~600[cent])
param[89]:CAf LFO2 TVF DEPTH ($00~$7F:0~2400[cent])
param[90]:CAf LFO2 TVA DEPTH ($00~$7F:0~100.0[%])

param[91]:PAf PITCH CONTROL ($28~$58:－24~＋24[semitone])
param[92]:PAf TVF CUTOFF CONTROL
($00~$7F:－9600~＋9600[CENT])
param[93]:PAf AMPLITUDE CONTROL
($00~$7F:－100.0~＋100.0[%])
param[94]:PAf LFO1 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[95]:PAf LFO1 PITCH DEPTH ($00~$7F:0~600[cent])
param[96]:PAf LFO1 TVF DEPTH ($00~$7F:0~2400[cent])
param[97]:PAf LFO1 TVA DEPTH ($00~$7F:0~100.0[%])
param[98]:PAf LFO2 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[99]:PAf LFO2 PITCH DEPTH ($00~$7F:0~600[cent])
param[100]:PAf LFO2 TVF DEPTH ($00~$7F:0~2400[cent])
param[101]:PAf LFO2 TVA DEPTH ($00~$7F:0~100.0[%])

param[102]:CC1 PITCH CONTROL ($28~$58:－24~＋24[semitone])
param[103]:CC1 TVF CUTOFF CONTROL
($00~$7F:－9600~＋9600[CENT])
param[104]:CC1 AMPLITUDE CONTROL
($00~$7F:－100.0~＋100.0[%])
param[105]:CC1 LFO1 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[106]:CC1 LFO1 PITCH DEPTH ($00~$7F:0~600[cent])
param[107]:CC1 LFO1 TVF DEPTH ($00~$7F:0~2400[cent])
param[108]:CC1 LFO1 TVA DEPTH ($00~$7F:0~100.0[%])
param[109]:CC1 LFO2 RATE CONTROL($00~$7F:－10.0~＋10.0[Hz])
param[110]:CC1 LFO2 PITCH DEPTH ($00~$7F:0~600[cent])
param[111]:CC1 LFO2 TVF DEPTH ($00~$7F:0~2400[cent])
param[112]:CC1 LFO2 TVA DEPTH ($00~$7F:0~100.0[%])

param[113]:CC2 PITCH CONTROL ($28~$58:－24~＋24[semitone])
param[114]:CC2 TVF CUTOFF CONTROL
($00~$7F:－9600~＋9600[CENT])
param[115]:CC2 AMPLITUDE CONTROL
($00~$7F:－100.0~＋100.0[%])
param[116]:CC2 LFO1 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[117]:CC2 LFO1 PITCH DEPTH ($00~$7F:0~600[cent])
param[118]:CC2 LFO1 TVF DEPTH ($00~$7F:0~2400[cent])
param[119]:CC2 LFO1 TVA DEPTH ($00~$7F:0~100.0[%])
param[120]:CC2 LFO2 RATE CONTROL
($00~$7F:－10.0~＋10.0[Hz])
param[121]:CC2 LFO2 PITCH DEPTH ($00~$7F:0~600[cent])
param[122]:CC2 LFO2 TVF DEPTH ($00~$7F:0~2400[cent])
param[123]:CC2 LFO2 TVA DEPTH ($00~$7F:0~100.0[%])
param[124]:TONE MAP NUMBER (0:SELECTED,1:SC-55,2:SC-88)
param[125]:TONE MAP0 NUMBER (1:SC-55,2:SC-88)
param[126]:EQ ON/OFF (0:OFF,1:ON)
(SC-88VLマニュアルp79~p81参照)

*int zm_sc88_drum_setup(int port, int size, int dr_id, char *param);*

機能 SC-88のドラムセットアップ設定
引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1-9)
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d23=NOTE NUMBER(0-127)
d24-d31=MAP NUMBER(0-1)
char *param:ドラムセットアップパラメータ格納バッファ
戻り値 0:正常終了
0以外:エラーコード
備考 char *paramからの領域にドラムセットアップパラメータを記述しておく。sizeは9以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
param[0]:PLAY KEY NUMBER (0~127)
param[1]:LEVEL (0~127)
param[2]:ASSIGN GROUP NUMBER (0:NON,1~127)
param[3]:PANPOT (0:RANDOM,01(LEFT)~64~127(RIGHT))
param[4]:REVERB DEPTH (0~127:0.0~1.0)
param[5]:CHORUS DEPTH (0~127:0.0~1.0)
param[6]:Rx NOTE OFF (0~1:OFF/ON)
param[7]:Rx NOTE ON (0~1:OFF/ON)
param[8]:DELAY SEND LEVEL (0-127:0.0~1.0)
(SC-88VLマニュアルp82参照)

*int zm_sc88_drum_name(int port, int size, int mp_id, char *name);*

機能 SC-88のドラムセット名設定
引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1-8)
int mp_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=MAP NUMBER(0,1,$80,$81)
char *name:ドラムセット名文字列格納バッファ
戻り値 0:正常終了
0以外:エラーコード
備考 文字列は12文字以内でchar *nameからの領域に文字列を記述しておく。idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。MAP NUMBER=$80がユーザードラムセット65,MAP NUMBER=$81がユーザードラムセット66に対応する。(SC-88VLマニュアルp82参照)

*int zm_sc88_user_inst(int port, int size, int ui_id, char *param);*

機能 SC-88のユーザー音色の設定
引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ユーザー音色パラメータデータサイズ(1-11)
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d23=PROGRAM NUMBER(0-127)
d24-d31=BANK NUMBER(0-1)
char *param:ユーザー音色パラメータ格納バッファ
戻り値 0:正常終了
0以外:エラーコード
備考 char *paramからの領域にユーザー音色・パラメータを記述しておく。sizeは11以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
param[0]:SOURCE TONE#(MAP) (1,2)
param[1]:BANK NUMBER MSB (0~127)
param[2]:PROGRAM NUMBER (0~127)
param[3]:Vibrato rate ($00~$7f:-64~+63)
param[4]:Vibrato depth ($00~$7f:-64~+63)

param[5]:TVF cutoff freq. ($00～$7f:-64～+63)
param[6]:TVF resonance ($00～$7f:-64～+63)
param[7]:TVF&TVA Env.attack ($00～$7f:-64～+63)
param[8]:TVF&TVA Env.decay ($00～$7f:-64～+63)
param[9]:TVF&TVA Env.release ($00～$7f:-64～+63)
param[10]:Vibrato delay ($00～$7f:-64～+63)
(SC-88VLマニュアルp82参照)

*int zm_sc88_user_drum(int port, int size, int ud_id, char *param);*

機能 SC-88のユーザードラムセットの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ユーザードラムセットパラメータデータサイズ(1-12)
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d23=NOTE NUMBER(0-127)
d24-d31=SET NUMBER(0-1)
char *param:ユーザードラムセットパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *paramからの領域にユーザードラムセットパラメータを記述しておく。
sizeは12以内で指定したsize個分のパラメータがSC-88へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
param[0]:PLAY KEY NUMBER (0～127)
param[1]:LEVEL (0～127)
param[2]:ASSIGN GROUP NUMBER (0:NON,1～127)
param[3]:PANPOT (0:RANDOM,01(LEFT)～64～127(RIGHT))
param[4]:REVERB DEPTH (0～127:0.0～1.0)
param[5]:CHORUS DEPTH (0～127:0.0～1.0)
param[6]:Rx NOTE OFF (0～1:OFF/ON)
param[7]:Rx NOTE ON (0～1:OFF/ON)
param[8]:DELAY SEND LEVEL (0-127:0.0～1.0)
param[9]:SOURCE DRUM SET#(MAP) (1-2)
param[10]:PROGRAM NUMBER (0-127)
param[11]:SOURCE NOTE NUMBER (0-127)
(SC-88VLマニュアルp82参照)

int zm_mt32_reset (int port, int id);

機能 MT-32の初期化

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)

戻り値 0:正常終了
0以外:エラーコード

備考 idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

*int zm_mt32_partial_reserve (int port, int id, char *rsv);*

機能 MT-32のパーシャルリザーブ

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *rsv:パート1～パート9までのパーシャルリザーブ値が格納されているバッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *rsvからの領域にパーシャルリザーブ値を9パート分格納しておく。また、合計が楽器の最大同時発声数以下にならなければならない。
param[0]=パート1,param[1]=パート2…,param[8]=パート9
というふうに対応する。なお、パート9はリズムパートに対応する。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

*int zm_mt32_reverb (int port, int size, int id, char *rsv);*

機能 MT-32のリバーブパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:リバーブパラメータデータサイズ(1-3)
int id:DEVICE ID(0-127,-1:以前の値を使用)
char *rvb:リバーブパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *rvbからの領域にリバーブパラメータを記述しておく。
sizeは3以内で、指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される
rvb[0]:REVERB MODE (0:Room,1:Hall,2:Plate,3:Tap Delay)
rvb[1]:REVERB TIME (0～7)
rvb[2]:REVERB LEVEL (0～7)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

*int zm_mt32_part_setup (int port, int size, int id, char *rsv);*

機能 MT-32のパートパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:パートパラメータデータサイズ(1-9)
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *param:パートパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *paramからの領域にパートパラメータを記述しておく。
パートパラメータは、
param[0]＝パート1のMIDIチャンネル,
param[1]＝パート2のMIDIチャンネル,
:
param[8]＝リズムパートのMIDIチャンネル
という構成を取る。
sizeは9以内で指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

*int zm_mt32_drum (int port, int size, int dr_id, char *drum);*

機能 MT-32のドラムセットアップ設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:ドラムセットアップパラメータデータサイズ(1-4)
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=NOTE NUMBER(24～87)
char *drum:ドラムセットアップパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *drumからの領域にドラムセットアップパラメータを記述しておく。
sizeは4以内で指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。
drum[0]:TIMBRE (0～63:i1～64,64～127:r1～64)
drum[1]:OUTPUT LEVEL (0～100)
drum[2]:PANPOT (0～14)
drum[3]:REVERB SWITCH (0～1:OFF/ON)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

*int zm_mt32_common(int port, int size, int cm_id, char *common);*

機能 MT-32のコモンパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:コモンパラメータデータサイズ(1-4)
int cm_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=PROGRAM NUMBER(1-64)
char *common:コモンパラメータ格納バッファ

戻り値 0:正常終了

0以外:エラーコード

備考 char *commonからの領域にコモンパラメータを記述しておく。sizeは4以内で指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

common[0]:Structure of Partial#1&2 (0〜12:1〜13)
common[1]:Structure of Partial#3&4 (0〜12:1〜13)
common[2]:PARTIAL MUTE (0〜15)
common[3]:ENV MODE (0〜1:NORMAL,NO SUSTAIN)
(MT-32マニュアルp34,CM-64マニュアルp29参照)

*int zm_mt32_partial(int port, int size, int pl_id, char *partial);*

機能 MT-32のパーシャルパラメータ設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:パーシャルパラメータデータサイズ(1-58)
int pl_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d23=PARTIAL NUMBER(1-4)
d24-d31=PROGRAM NUMBER(1-64)
char *partial:パーシャル・パラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *partialからの領域にパーシャルパラメータを記述しておく。
sizeは58以内で指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

partial[0]:WG PITCH COARSE (0〜96:C1,C#1,〜,C9)
partial[1]:WG PITCH FINE (0〜100:-50〜+50)
partial[2]:WG PITCH KEYFOLLOW (0〜16:−1,−1/2,−1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2,s1,s2)
partial[3]:WG PITCH BENDER SW (0〜1:OFF/ON)
partial[4]:WG WAVEFORM/PCM BANK (0〜3:SQU1,SAW1,SQU2,SAW2)
partial[5]:WG PCM WAVE (0〜127:1〜128)
partial[6]:WG PULSE WIDTH (0〜100)
partial[7]:WG PW VELO SENS (0〜14:-7〜+7)
partial[8]:P-ENV DEPTH (0〜10)
partial[9]:P-ENV VELO SENS (0〜3)
partial[10]:P-ENV TIME KEYF (0〜4)
partial[11]:P-ENV TIME 1 (0〜100)
partial[12]:P-ENV TIME 2 (0〜100)
partial[13]:P-ENV TIME 3 (0〜100)
partial[14]:P-ENV TIME 4 (0〜100)
partial[15]:P-ENV LEVEL 0 (0〜100:−50〜+50)
partial[16]:P-ENV LEVEL 1 (0〜100:−50〜+50))
partial[17]:P-ENV LEVEL 2 (0〜100:−50〜+50)
partial[18]:P-ENV SUSTAIN LEVEL (0〜100:−50〜+50)
partial[19]:END LEVEL (0〜100:−50〜+50)
partial[20]:P-LFO RATE (0〜100)
partial[21]:P-LFO DEPTH (0〜100)
partial[22]:P-LFO MOD SENS (0〜100)
partial[23]:TVF CUTOFF FREQ (0〜100)
partial[24]:TVF RESONANCE (0〜30)
partial[25]:TVF KEYFOLLOW (0〜14:−1,−1/2,−1/4,0,1/8,1/4,3/8,1/2,5/8,3/4,7/8,1,5/4,3/2,2)
partial[26]:TVF BIAS POINT/DIR (0〜127)
partial[27]:TVF BIAS LEVEL (0〜14:-7〜+7)
partial[28]:TVF ENV DEPTH (0〜100)
partial[29]:TVF ENV VELO SENS (0〜100)
partial[30]:TVF ENV DEPTH KEYF (0〜4)
partial[31]:TVF ENV TIME KEYF (0〜4)
partial[32]:TVF ENV TIME 1 (0〜100)
partial[33]:TVF ENV TIME 2 (0〜100)
partial[34]:TVF ENV TIME 3 (0〜100)
partial[35]:TVF ENV TIME 4 (0〜100)
partial[36]:TVF ENV TIME 5 (0〜100)
partial[37]:TVF ENV LEVEL 1 (0〜100)
partial[38]:TVF ENV LEVEL 2 (0〜100)
partial[39]:TVF ENV LEVEL 3 (0〜100)
partial[40]:TVF ENV SUSTAIN LEVEL (0〜100)
partial[41]:TVA LEVEL (0〜100)
partial[42]:TVA VELO SENS (0〜100:−50〜+50)
partial[43]:TVA BIAS POINT 1 (0〜127)
partial[44]:TVA BIAS LEVEL 1 (0〜12:−12〜0)
partial[45]:TVA BIAS POINT 2 (0〜127)
partial[46]:TVA BIAS LEVEL 2 (0〜12:−12〜0)
partial[47]:TVA ENV TIME KEYF (0〜4)
partial[48]:TVA ENV TIME V_FOLLOW (0〜4)
partial[49]:TVA ENV TIME 1 (0〜100)
partial[50]:TVA ENV TIME 2 (0〜100)
partial[51]:TVA ENV TIME 3 (0〜100)
partial[52]:TVA ENV TIME 4 (0〜100)
partial[53]:TVA ENV TIME 5 (0〜100)
partial[54]:TVA ENV LEVEL 1 (0〜100)
partial[55]:TVA ENV LEVEL 2 (0〜100)
partial[56]:TVA ENV LEVEL 3 (0〜100)
partial[57]:TVA ENV SUSTAIN LEVEL (0〜100)
(MT-32マニュアルp34,CM-64マニュアルp29参照)

*int zm_mt32_patch(int port, int size, int pt_id, char *patch);*

機能 MT-32のパッチパラメータ設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:パッチパラメータデータサイズ(1-7)
int pt_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=PATCH NUMBER(1-128)
char *patch:パッチパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *patchからの領域にパッチパラメータを記述しておく。
sizeは7以内で指定したsize個分のパラメータがMT-32へ送信される。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

patch[0]:TIMBRE GROUP (0〜3:a,b,i,r)
patch[1]:TIMBRE NUMBER (0〜63)
patch[2]:KEY SHIFT (0〜48:−24〜+24)
patch[3]:FINE TUNE (0〜100:−50〜+50)
patch[4]:BENDER RANGE (0〜24)
patch[5]:ASSIGN MODE (0〜3:POLY1〜4)
patch[6]:REVERB SWITCH (0〜1:OFF/ON)
(MT-32マニュアルp35,CM-64マニュアルp30参照)

*int zm_mt32_print(int port, int size, int id, char *mes);*

機能 MT-32の画面へのメッセージ出力

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int size:メッセージ長(1-20)
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *mes:文字列格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 メッセージは20文字以内でchar *mesからの領域に文字列を記述しておく。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

*int zm_u220_setup(int port, int id, char *setup);*

機能 U220のセットアップパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *setup:セットアップパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *setupからの領域にセットアップパラメータを記述しておく。
char *setupの構成は必ず下に示す書式に従う。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
setup[0]:Chorus SW          (0～1:OFF/ON)
setup[1]:Reverb SW          (0～1:OFF/ON)
setup[2]:Rx Cntrl Channel   (1～17:1～16,OFF)
setup[3]:Patch Change       (0～5:MAP1～MAP4,Dir,OFF)
setup[4]:Timbre Change      (0～5:MAP1～MAP4,Dir,OFF)
setup[5]:Rhythm Change      (0～5:MAP1～MAP4,Dir,OFF)
setup[6]:R.Inst Assign      (0～5:MAP1～MAP4,Dir,OFF)
```

(U220マニュアルp46,p146～148参照)

注意 U220用の関数はU20に対しても使用できる。以下同様。

int zm_u220_part_setup(int port, int pt_id, char *setup);

機能 U220の(テンポラリパッチ)パートセットアップパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-31=PART NUMBER(1-6)
char *setup:パートセットアップパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *setupからの領域にパートセットアップパラメータを記述しておく。
char *setupの構成は必ず下に示す書式に従う。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
setup[0]:Timbre Number      (1～128)
setup[1]:Voice Reserve      (0～30)
setup[2]:Receive Channel    (1～17:1～16,OFF)
setup[3]:Key Range Low      (0～127:C-1～G9)
setup[4]:Key Range High     (0～127:C-1～G9)
setup[5]:Velo Level         (0～1:Above/Below)
setup[6]:Velo Threshold     (0～127)
setup[7]:Output Assign      (0～4:Dry,Rev,Cho,Dir1,Dir2)
setup[8]:Level              (0～127)
setup[9]:Pan                (0～15:L7～M～R7)
setup[10]:Rx Volume         (0～1:OFF/ON)
setup[11]:Rx Pan            (0～1:OFF/ON)
setup[12]:Rx Hold           (0～1:OFF/ON)
```

(U220マニュアルp58,p149参照)

int zm_u220_common(int port, int id, char *common);

機能 U220の(テンポラリパッチ)コモンパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *common:コモンパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *commonからの領域にコモンパラメータを記述しておく。
char *commonの構成は必ず下に示す書式に従う。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
common[0]:Chorus Type        (0～4:Chorus1,Chorus2,FB-Chorus,
                              Flanger,Short Delay)
common[1]:Chorus Out Mode    (0～1:Pre Rev,Post Rev)
common[2]:Chorus Level       (0～31)
common[3]:Chorus Delay       (0～31)
common[4]:Chorus Rate        (0～31)
common[5]:Chorus Depth       (0～31)
common[6]:Chorus Feedback    (1～63:－31～0～＋31)
common[7]:Reverb Type        (0～7:Room1～3,Hall1～2,
                              GateDelay,CrossDelay)
common[8]:Reverb Time        (0～31)
common[9]:Reverb Level       (0～31)
common[10]:Reverb Delay Feedback   (0～31)
common[11]:Reverb Pre Delay Feedback  (0～31)
common[12]:Param1 #          (0～63:0～5,7～31,64～95,OFF)
common[13]:Param1 Param  (0:Timbre Level,1:Env Attack
                          2:Env Decay,3:Env Sustain
                          4:Env Release,5:A.Bend Depth
                          6:A.Bend Rate,7:Detune Depth
                          8:Vib Rate,9:Vib Wave Form
                          10:Vib Depth,11:Vib Delay
                          12:Vib Rise Time,13:Vib Mod Depth
                          14:Chorus Level,15:Chorus Rate
                          16:Chorus Feedback,17:ReverbLevel
                          18:Delay Feedback)
common[14]:Param2 #          (Same as Param1 #)
common[15]:Param2 Param      (Same as Param1 Param)
common[16]:Param3 #          (Same as Param1 #)
common[17]:Param3 Param      (Same as Param1 Param)
```

(U220マニュアルp54,p149参照)

int zm_u220_timbre(int port, int tm_id, char *timbre);

機能 U220のティンバーパラメータの設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int tm_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
d16-d31=PROGRAM NUMBER(1-128)
char *timbre:ティンバーパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *timbreからの領域にティンバー・パラメータを記述しておく。char *timbreの構成は必ず下に示す書式に従う。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
timbre[0]:Tone Media            (0～31:I,1～31)
timbre[1]:Tone Number           (1～128)
timbre[2]:Timbre Level          (0～127)
timbre[3]:Velocity Sens         (1～15:-7～+7)
timbre[4]:Channel Press Sens    (1～15:-7～+7)
timbre[5]:Env Attack Rate       (1～15:-7～+7)
timbre[6]:Env Decay Rate        (1～15:-7～+7)
timbre[7]:Env Sustain Level     (1～15:-7～+7)
timbre[8]:Env Release Rate      (1～15:-7～7)
timbre[9]:Pitch Shift Coarse    (8～56:-24～+24)
timbre[10]:Pitch Shift Fine     (14～114:－50～＋50)
timbre[11]:Bend Range Lower     (0～15:－36,－24,－12～0)
timbre[12]:Bend Range Upper     (0～12)
timbre[13]:Channel After Sens   (0～27:－36,－24,－12～＋12)
timbre[14]:Poly After Sens      (0～27:－36,－24,－12～＋12)
timbre[15]:Auto Bend Depth      (0～27:－36,－24,－12～＋12)
timbre[16]:Auto Bend Rate       (0～15)
timbre[17]:Detune Depth         (0～15)
timbre[18]:Rate                 (0～63)
timbre[19]:Waveform             (0～8)
timbre[20]:Depth                (0～15)
timbre[21]:Delay                (0～15)
timbre[22]:Rise Time            (0～15)
timbre[23]:Modulation Depth     (0～15)
timbre[24]:Ch After Sens        (0～15)
timbre[25]:Poly After Sens      (0～15)
```

(U220マニュアルp65,p149参照)

int zm_u220_drum_setup(int port, int id, char *setup);

機能 U220の(テンポラリパッチ)パッチドラムセットアップ設定

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *setup:ドラムセットアップパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *setupからの領域にドラムセットアップパラメータを記述しておく。char *setupの構成は必ず下に示す書式に従う。

idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
setup[0]:Rhythm Setup #        (0〜3:1〜4)
setup[1]:Voice Reserve         (0〜30)
setup[2]:Receive Channel       (1〜17:1〜16,OFF)
setup[3]:Level                 (0〜127)
setup[4]:Level Boost Sw        (0〜1:OFF/ON)
setup[5]:Rx Volume             (0〜1:OFF/ON)
setup[6]:Rx Hold               (0〜1:OFF/ON)
```

(U220マニュアルp63,p149参照)

int zm_u220_drum_inst(int port, int dr_id, char *inst);

機能 U220の(テンポラリパッチ)ドラム音色設定

引数 int port:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
int dr_id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
　:d16-d31=NOTE NUMBER(35-99)
char *inst:ドラム音色パラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *instからの領域にドラム音色パラメータを記述しておく。
char *instの構成は必ず下に示す書式に従う。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

```
inst[0]:Tone Media             (0〜31:I,1〜31)
inst[1]:Tone Number            (1〜128)
inst[2]:Source Key             (0〜127:C－1〜G9)
inst[3]:Mute Inst              (34〜98:OFF,B1〜D7)
inst[4]:Inst Level             (0〜31)
inst[5]:Velocity Sens          (0〜15)
inst[6]:Env Mode               (0〜1:Sustain,No Sustain)
inst[7]:Env Attack Rate        (1〜15:－7〜＋7)
inst[8]:Env Decay Rate         (1〜15:－7〜＋7)
inst[9]:Env Release Rate       (1〜15:－7〜＋7)
inst[10]:Pitch Shift Coarse    (0〜27:－36,－24,－12〜＋12)
inst[11]:Pitch Shift Fine      (14〜114:－50〜＋50)
inst[12]:Channel After Sens    (0〜27:－36,－24,－12〜＋12)
inst[13]:Poly After Sens       (0〜27:－36,－24,－12〜＋12)
inst[14]:Random                (0〜15)
inst[15]:Auto Bend Depth       (0〜27:－36,－24,－12〜＋12)
inst[16]:Auto Bend Rate        (0〜15)
inst[17]:Detune Depth          (0〜15)
inst[18]:Output Assign         (0〜3:Dry,Rev,Cho,Dir1)
inst[19]:Pan                   (0〜15:L7〜M〜R7)
```

(U220マニュアルp72,p149)

int zm_u220_print(int port, int size, int id, char *mes);

機能 U220の画面へのメッセージ出力

引数 int port:MIDIポートの番号(0-3)
　-1はカレントMIDI-OUTを使用
int size:メッセージ長(1-12)
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
char *mes:文字列格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 メッセージは12文字以内でchar *mesからの領域に文字列を記述しておく。
idを-1とすると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$10が選択される。

int zm_m1_setup(char *setup);

機能 M1のSEQ-SONG0のMIDIチャンネルの設定

引数 char *setup:8パート分のMIDIチャンネル列格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *setupで表される領域には必ず8パート分のMIDIチャンネルを記載しておく。

```
setup[0]:パート1のMIDIチャンネル
setup[1]:パート2のMIDIチャンネル
setup[2]:パート3のMIDIチャンネル
setup[3]:パート4のMIDIチャンネル
setup[4]:パート5のMIDIチャンネル
setup[5]:パート6のMIDIチャンネル
setup[6]:パート7のMIDIチャンネル
setup[7]:パート8のMIDIチャンネル
```

注意 M1用の命令はM1EX/M1R/M1REXに対しても使用できる。以下同様。

int zm_m1_part_setup(char *ptstup);

機能 M1のSEQ-SONG0のトラックパラメータの設定

引数 char *ptstup:8パート分のトラックパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *ptstupで表される領域には必ず8パート分のトラックパラメータを記載しておく。

```
TRACK 1
ptstup[0]:PROGRAM NUMBER       (0〜199:100〜C99)
ptstup[1]:OUTPUT LEVEL         (0〜99)
ptstup[2]:KEY TRANSPOSE        ($F4〜$0C:－12〜12)
ptstup[3]:DETUNE               ($CE〜$32:－50〜＋50)
ptstup[4]:PAN                  (0〜13:10:0〜0:10,C,C+D,D)
TRACK 2〜8
以下同様
```

(M1マニュアルp126参照)

int zm_m1_effect_setup(char *effect);

機能 M1のSEQ-SONG0のエフェクトパラメータの設定

引数 char *effect:エフェクトパラメータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *effectで表される領域にはダミーデータも含めて必ず25個のエフェクトパラメータが記載されていなければならない。

```
effect[0]:Effect 1 Pattern No.     (0〜33:1〜32,Thru)
effect[1]:Effect 2 Pattern No.     (0〜33:1〜32,Thru)
effect[2]:Effect 1 L-Ch E.Balnc    (0〜100)
effect[3]:Effect 1 R-Ch E.Balnc    (0〜100)
effect[4]:Effect 2 L-Ch E.Balnc    (0〜100)
effect[5]:Effect 2 R-Ch E.Balnc    (0〜100)
effect[6]:Output 3 Pan             (0〜101:OFF,100:0〜0:100)
effect[7]:Output 4 Pan             (0〜101:OFF,100:0〜0:100)
effect[8]:Effect I/O (BIT0:Effect 1 L-Ch 0＝OFF/1＝ON
                      BIT1:Effect 1 R-Ch 0＝OFF/1＝ON
                      BIT2:Effect 2 L-Ch 0＝OFF/1＝ON
                      BIT3:Effect 2 R-Ch 0＝OFF/1＝ON)
                      BIT4:Effect 2 0＝PARALLEL/1＝SERIAL)
effect[9〜16]:Effect 1 Parameter
effect[17〜24]:Effect 2 Parameter
```

Structure of Effect Parameter

(ofs=9 or 17)

1〜3:Hall (4,5:Room,6:Live Stage)

```
effect[ofs+0]:Reverb Time (0〜97:0.2〜9.9),(0〜48:0.2〜5.0)
effect[ofs+1]:Dummy            0
effect[ofs+2]:High Damp        (0〜99)
effect[ofs+3]:Pre Delay        (0〜200)
effect[ofs+4]:E/R Level        (0〜99)
effect[ofs+5]:Dummy            0
effect[ofs+6]:EQ High          ($F4〜$0C:－12〜＋12)
effect[ofs+7]:EQ Low           ($F4〜$0C:－12〜＋12)
```

7〜9:Early Reflection 1,2,3

```
effect[ofs+0]:E/R Time         (0〜70:100〜800)
effect[ofs+1]:Pre Delay        (0〜200)
effect[ofs+2]:Dummy            0
      :                        :
effect[ofs+5]:Dummy            0
effect[ofs+6]:EQ High          ($F4〜$0C:－12〜＋12)
effect[ofs+7]:EQ Low           ($F4〜$0C:－12〜＋12)
```

10:Stereo Delayt,11:Cross Delay

effect[ofs+0]:Delay Time L (L)	(0〜500)
effect[ofs+1]:Delay Time L (H)	
effect[ofs+2]:Feedback	($9D〜$63:−99〜99)
effect[ofs+3]:High Damp	(0〜99)
effect[ofs+4]:Delay Time R (L)	(0〜500)
effect[ofs+5]:Delay Time R (H)	
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

12〜13:Stereo Chorus 1〜2 (14〜15:Flanger)

effect[ofs+0]:Depth	(0〜99)
effect[ofs+1]:Speed	(0〜99:0.03〜3.00 0.03step 100〜199:3.1〜13.0 0.1step 200〜216:14.0〜30.0 1.0step)
effect[ofs+2]:LFO Status	(BIT0:Waveform =0:Sin,=1:Tri BIT1:Phase =0:0.0°,=1:180° BIT2:Wave Shape=0:Normal =1:for Flanger)
effect[ofs+3]:Feedback	($9D〜$63:−99〜+99)
effect[ofs+4]:Delay Time	(0〜200),(0〜50)
effect[ofs+5]:Dummy	0
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

16〜17:Phase Shifter 1〜2

effect[ofs+0]:Depth	(0〜99)
effect[ofs+1]:Speed	(0〜99:0.03〜3.00 0.03step 100〜199:3.1〜13.0 0.1step 200〜216:14.0〜30.0 1.0step)
effect[ofs+2]:LFO Status	(BIT0:Waveform =0:Sin,=1:Tri BIT1:Phase =0:0.0°,=1:180° BIT2:Wave Shape =0:Normal =1:for Flanger)
effect[ofs+3]:Feedback	($9D〜$63:−99〜+99)
effect[ofs+4]:Manual	(0〜99)
effect[ofs+5]:Dummy	0
effect[ofs+6]:Dummy	0
effect[ofs+7]:Dummy	0

18〜19:Stereo Toremolo 1〜2

effect[ofs+0]:Depth	(0〜99)
effect[ofs+1]:Speed	(0〜99:0.03〜3.00 0.03step 100〜199:3.1〜13.0 0.1step 200〜216:14.0〜30.0 1.0step)
effect[ofs+2]:LFO Status	(BIT0:Waveform =0:Sin,=1:Tri BIT1:Phase =0:0.0°,=1:180° BIT2:Wave Shape=0:Normal =1:for Flanger)
effect[ofs+3]:Shape	($9D〜$63:−99〜+99)
effect[ofs+4]:Dummy	0
effect[ofs+5]:Dummy	0
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

20:Equalizer

effect[ofs+0]:Dummy	0
:	:
effect[ofs+3]:Dummy	0
effect[ofs+4]:Low fc	(0〜2:0.25k,0.50k,1.00k)
effect[ofs+5]:High fc	(0〜2:1k,2k,4k)
effect[ofs+6]:High Gain	($F4〜$0C:−12〜+12)
effect[ofs+7]:Low Gain	($F4〜$0C:−12〜+12)

21:Overdrive

effect[ofs+0]:Dummy	0
effect[ofs+1]:Dummy	0
effect[ofs+2]:Drive	(0〜99)
effect[ofs+3]:Level	(0〜99)
effect[ofs+4]:Dummy	0
effect[ofs+5]:Dummy	0
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

22:Distortion

effect[ofs+0]:Dummy	0
effect[ofs+1]:Dummy	0
effect[ofs+2]:Distortion	(0〜99)
effect[ofs+3]:Level	(0〜99)
effect[ofs+4]:Dummy	0
effect[ofs+5]:Dummy	0
effect[ofs+6]:Dummy	0
effect[ofs+7]:EQ Low Gain	($F4〜$0C:−12〜+12)

23:Exciter

effect[ofs+0]:Blend	($9D〜$63:−99〜+99)
effect[ofs+1]:Emphatic Point	(0〜9:1〜10)
effect[ofs+2]:Dummy	0
:	:
effect[ofs+5]:Dummy	0
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

24:Synphonic Ensamble

effect[ofs+0]:Depth	(0〜99)
effect[ofs+1]:Dummy	0
:	:
effect[ofs+5]:Dummy	0
effect[ofs+6]:EQ High	($F4〜$0C:−12〜+12)
effect[ofs+7]:EQ Low	($F4〜$0C:−12〜+12)

25:Roteffect Speaker

effect[ofs+0]:Depth	(0〜99)
effect[ofs+1]:Dummy	0
effect[ofs+2]:Speed Rate	($F6〜$0A:−10〜+10)
effect[ofs+3]:Dummy	0
:	:
effect[ofs+7]:Dummy	0

26:Delay/Hall

effect[ofs+0]:Delay Time (L)	(0〜500)
effect[ofs+1]:Delay Time (H)	
effect[ofs+2]:Feedback	($9D〜$63:−99〜99)
effect[ofs+3]:High Damp	(0〜99)
effect[ofs+4]:Reverb Time	(0〜97:0.2〜9.9)
effect[ofs+5]:Dummy	0
effect[ofs+6]:High Damp	(0〜99)
effect[ofs+7]:Pre Delay	(0〜150)

27:Delay/Room

effect[ofs+0]:Delay Time (L)	(0〜500)
effect[ofs+1]:Delay Time (H)	
effect[ofs+2]:Feedback	($9D〜$63:−99〜99)
effect[ofs+3]:High Damp	(0〜99)
effect[ofs+4]:Reverb Time	(0〜97:0.2〜9.9)
effect[ofs+5]:Dummy	0
effect[ofs+6]:High Damp	(0〜99)
effect[ofs+7]:Pre Delay	(0〜150)

28:Delay/Early Reflection

effect[ofs+0]:Delay Time (L)	(0〜500)
effect[ofs+1]:Delay Time (H)	
effect[ofs+2]:Feedback	($9D〜$63:−99〜99)
effect[ofs+3]:High Damp	(0〜99)
effect[ofs+4]:E/R Time	(0〜30:100〜400)
effect[ofs+5]:Pre Delay	(0〜150)
effect[ofs+6]:Dummy	0
effect[ofs+7]:Dummy	0

29:Delay/Chorus

effect[ofs+0]:Delay Time L (L)　(0～500)
effect[ofs+1]:Delay Time L (H)
effect[ofs+2]:Feedback L　($9D～$63:－99～99)
effect[ofs+3]:High Damp L　(0～99)
effect[ofs+4]:Delay Time R (L)　(0～500)
effect[ofs+5]:Delay Time R (H)
effect[ofs+6]:Feedback R　($9D～$63:－99～99)
effect[ofs+7]:High Damp R　(0～99)

30:Delay/Chorus (31:Delay Flanger)
effect[ofs+0]:Delay Time (L)　(0～500)
effect[ofs+1]:Delay Time (H)
effect[ofs+2]:Feedback　($9D～$63:－99～99)
effect[ofs+3]:High Damp　(0～99)
effect[ofs+4]:Depth　(0～99)
effect[ofs+5]:Speed　(0～99:0.03～3.00 0.03step
100～199:3.1～13.0 0.1step
200～216:14.0～30.0 1.0step)
effect[ofs+6]:LFO Status (BIT0:Waveform =0:Sin,=1:Tri
BIT1:Phase =0:0.0°,=1:180°
BIT2:Wave Shape =0:Normal
=1:for Flanger)
effect[ofs+7]:Feedback　0,($9D～$63:－99～99)

32:Delay/Phaser
effect[ofs+0]:Delay Time (L)　(0～500)
effect[ofs+1]:Delay Time (H)
effect[ofs+2]:Feedback　($9D～$63:－99～99)
effect[ofs+3]:High Damp　(0～99)
effect[ofs+4]:Depth　(0～99)
effect[ofs+5]:Speed　(0～99:0.03～3.00 0.03step)
effect[ofs+6]:Feedback　($9D～$63:－99～99)
effect[ofs+7]:Dummy　0

33:Delay/Tremolo
effect[ofs+0]:Delay Time (L)　(0～500)
effect[ofs+1]:Delay Time (H)
effect[ofs+2]:Feedback　($9D～$63:－99～99)
effect[ofs+3]:High Damp　(0～99)
effect[ofs+4]:Depth　(0～99)
effect[ofs+5]:Speed　(0～99:0.03～3.00 0.03step)
effect[ofs+6]:Dummy　0
effect[ofs+7]:Shape　($9D～$63:－99～99)
(M1マニュアルp127参照)

*int zm_m1_print(int size, char *mes);*

機能　M1のSEQ-SONG0のタイトル設定
引数　int size:メッセージ長(1-10)
char *mes:文字列格納バッファ
戻り値　0:正常終了
0以外:エラーコード
備考　メッセージは10文字以内でchar *mesからの領域に文字列を記述しておく。

int zm_send_to_m1(int port, int id);

機能　M1へパラメータを送信する
引数　int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
int id:d0-d15=DEVICE ID(0-127,-1:以前の値を使用)
戻り値　0:正常終了
0以外:エラーコード
備考　DEVICE ID=&H30+(グローバルチャンネル－1)
グローバルチャンネルはGLOBALモードのF5－1で設定できる。
idは省略可能。省略すると以前設定したものが選択される。最初の使用時に省略した場合はZ-MUSIC内のデフォルト値$30が選択される。
必ず'm1_setup()'～'m1_print()'を設定してからこの命令を実行すること。'm1_effect_setup()'のみ省略が可能。このときはZ-MUSICが持つデフォルトデータが送信される。
'm1_setup()'～'m1_print()'の間にＭ１関係以外のコマンドがあると正常なデータが送信されない。

例
```
char a[8]={2,1,4,3,6,5,8,7};
/*音色,音量,ｷｰﾄﾗﾝｽﾎﾟｰｽﾞ,ﾃﾞﾁｭｰﾝ,ﾊﾟﾝのﾌｫｰﾏｯﾄ*/
char b[40]={  0,99,0,0,5,  /*TRACK 1*/
              0,99,0,0,5,  /*TRACK 2*/
              0,99,0,0,5,  /*TRACK 3*/
              0,99,0,0,5,  /*TRACK 4*/
              0,99,0,0,5,  /*TRACK 5*/
              0,99,0,0,5,  /*TRACK 6*/
              0,99,0,0,5,  /*TRACK 7*/
              0,99,0,0,5}; /*TRACK 8*/
m1_setup(a);
m1_part_setup(b);
m1_print(7,"Zenji.N");
send_to_m1(-1,&H30);
```

*char *zm_pcm_read(int num, int flag, int type, int orig, char *param);*

機能　(AD)PCMファイルの登録
引数　int num:登録番号(トーン番号の場合:0～32767)
(音色番号の場合:$8000+(0～32767)
int flag:(AD)PCM加工コマンド(PPC)で(AD)PCMデータ加工するか
(-1:Yes,0:No)
int type:登録次元(-1:AD PCM,1=16BitPCM,0=V2互換)
int orig:オリジナルキー:0-127
char *param:登録データ情報およびPPC列格納バッファ(備考参照)
戻り値　0以外:char *param処理後のアドレス
0:エラー
備考　char *paramからはまず、最大32文字分の音色名文字列を格納する。終端コードは0。
次にディスクから(AD)PCMファイルを読みこむ場合はそのファイル名を格納する。終端コードは0。...(1)
ディスクからデータを読み込まずにすでにZ-MUSIC側に登録されている(AD)PCMデータをコピーして加工するには音色名の後に$0000 を格納し、さらにその後に参照番号を指定する。...(2)
PPCで(AD)PCMを加工する場合には、PPC列をこの後ろに格納しておく。
また、PPCで(AD)PCMを加工する場合は必ずint flagを-1とすること。flagが0では、たとえPPCがあっても無視される。
PPCについてはMEASURE12を参照。

(1)の場合
[TONE NAME...最大32文字],0,[FILENAME文字列],0,PPCコマンド列...
(2)の場合
[TONE NAME...最大32文字],0,$00,$00,参照先番号(2bytes),PPCｺﾏﾝﾄﾞ列...
参照先番号は、
ノート番号の場合:0～32767
音色番号の場合:$8000+(0～32767)
で表される。

int zm_pcm_erase(int num);

機能　(AD)PCMファイルの登録取消
引数　int num:取消したい登録番号(ノート番号の場合:0～32767)
(音色番号の場合:$8000+(0～32767)
戻り値　0:正常終了
0以外:エラーコード

*int zm_register_zpd(char *zpd);*

機能　ZPDの登録
引数　char *zpd:登録するZPDのファイルネーム
戻り値　0:正常終了
0以外:エラーコード

*int zm_set_zpd_table(int ver, char *zpd);*

機能　ZPDテーブルの登録
引数　int ver:ZPDバージョンタイプ(0:V1.0～V2.0,1:V3,2:PDX)
char *zpd:登録するZPDテーブル
(ZPDヘッダ「$1a,'ZmaDPcM'」含まず)
戻り値　0:正常終了

0以外:エラーコード

備考 ZPDのテーブルの構造についてはMEASURE12を参照

*int zm_exec_subfile(char *filename);*

機能 サブファイルの読み込みと実行

引数 char *filename:サブファイルのファイル名

戻り値 0:正常終了
0以外:エラーコード

備考 サブファイルとは演奏の際に必要となる別ファイルにまとめた補助ファイルのことを指す。具体的には、演奏ファイル(ZMS,ZMD,OPM)、AD PCM定義リスト(CNF),ZPDファイル、MIDIデータファイル(MDD,MID)など。

*int zm_transmit_midi_dump(int port, char *filename);*

機能 MIDIデータファイルの転送

引数 int port:MIDIポートの番号(0-3)
-1はカレントMIDI-OUTを使用
char *filename:MIDIデータファイルのファイル名

戻り値 0:正常終了
0以外:エラーコード

備考 MIDIデータファイルにはver.2.0以前で標準MIDIデータファイルとされてきたMDDファイルとver.3.0から採用されたスタンダードMIDIファイル(FORMAT0)の2タイプがあるが、この関数ではどちらもMIDI機器へ転送することができる。

*int zm_set_wave_form1(int wave, char *data);*

機能 波形メモリの登録1

引数 int wave:波形番号(8-32767)
char *data:波形メモリデータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *dataで表される波形メモリデータは以下の構成を取る。
data+0(int):波形のデータサイズ(必ず偶数)
data+4(char):ループタイプ(0:oneshot,1:repeat,2:altanative)
data+5(int):ループ開始オフセット
data+9(int):ループ終了オフセット
data+13(int):ループ回数(0:∞,1〜)
data+17(int):ダミー
data+21(char):波形名文字列長
data+22(?bytes):文字列(終端コード0無し)
偶数アドレスから実際の波形データ(signed short)
:
:
※本関数で登録された波形はZ-MUSICに取り込まれるため、関数実行後、波形データ本体のメモリを解放しても構わない。

*int zm_set_wave_form2(int wave, char *data);*

機能 波形メモリの登録2

引数 int wave:波形番号(8-32767)
char *data:波形メモリデータ格納バッファ

戻り値 0:正常終了
0以外:エラーコード

備考 char *dataで表される波形メモリデータは以下の構成は zm_set_wave_form1()と同じ。
※本関数で登録された波形はZ-MUSICに取り込まれないため、関数実行後、波形データ本体のメモリを解放してはならない。

*int *zm_obtain_events(int omt, int *event);*

機能 各種イベントの取得

引数 int omt:どのイベントに関する設定を行うかのビットマーカー
ビット=1でそのイベントの設定を更新する。
int event(8):各種イベント発生時のジャンプ先とジャンプ条件が記述された配列。構成は備考参照。
-1とするとint omtによらず設定は一切行わない。

戻り値 現在の設定を格納した領域のアドレス。構成は備考参照。

備考 event(0):m_playが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ　omtのd0ビット
event(1):m_stopが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ　omtのd1ビット
event(2):m_contが実行時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ　omtのd2ビット
event(3):演奏が終了時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ　omtのd3ビット
event(4):指定ループ回数(0:OFF,1-256)　omtのd4ビット
event(5):指定回数ﾙｰﾌﾟ時のｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ　omtのd5ビット
event(6):指定クロック(0:OFF,1〜)　omtのd6ビット
event(7):演奏開始後指定クロック経過時のジャンプ先アドレス　omtのd7ビット
int omtでビット=1としたパラメータのみをint event(0)から記述しておく。int omtでビット=0としたパラメータに関しては記述しない。
たとえばint omt=$81の場合、
event(0)に「m_playが実行されたときのジャンプ先アドレス」
event(1)に「奏開始後指定ｸﾛｯｸ経過したときのｼﾞｬﾝﾌﾟ先ｱﾄﾞﾚｽ」
を記述しておけばよい。

int zm_loop_control(int mode);

機能 現在の演奏のループを制御する

引数 int mode:ループ制御モード
-1=現在のループ回数を得る
0=通常ループモード
1=ループしない

戻り値 -1以外:現在のループ回数(1〜)
-1:エラー

*short *zm_mask_tracks(short *track);*

機能 トラックマスク

引数 short *track:マスクするトラックとマスクモードのペアを格納したバッファ

戻り値 設定できなかったトラック番号(short int)の並びを格納した領域のアドレス
終端コード=-1(short)

備考 short *trackからは
(short)トラック番号:0-65534,
(short)マスクモード(0:MASK OFF,1:REVERSE,-1:MASK ON)
の数値列を記載する。終端コードは(short)-1。

int zm_mask_all_tracks(int mode);

機能 全トラックマスク

引数 int mode:(0:MASK OFF,1:REVERSE,-1:MASK ON)

戻り値 0:正常終了
0以外:エラーコード

int zm_solo_track(int track);

機能 ソロトラック演奏

引数 int track:トラック番号(0-65534)

戻り値 0:正常終了
0以外:エラーコード

*int *zm_mask_channels(short *channel);*

機能 チャンネルマスク

引数 short *channel:マスクするチャンネルIDとマスクモードのペアを格納したアドレス

戻り値 設定できなかった(int)チャンネルIDの並びを格納したアドレス
終端コード=(int)-1

備考 int *channelからは、
(int)チャンネルID,(short)マスクモード
の数値列を記載する。なお、チャンネルIDは、
チャンネルID(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI
チャンネルID(下位16ビット):チャンネル番号0-15
マスクモードは(0:MASK OFF,1:REVERSE,-1:MASK ON)の値をとる。
終端コードは(short)-1。
たとえばMIDI1の15チャンネル(ID:$8000_000e)をﾏｽｸ(ID:$ffff)、AD PCMの3チャンネル(ID:$0001_0002)をマスク反転(ID:$0001)する場合は、
{ 0x8000,0x000e,0xffff,0x0001,0x0002,0x0001,0xffff}
となる。

void zm_mask_all_channels(int mode);
機能 全チャンネルマスク
引数 int mode:(0:MASK OFF,1:REVERSE,-1:MASK ON)
戻り値 0:正常終了
0以外:エラーコード

void zm_solo_channel(int channel);
機能 ソロチャンネル演奏
引数 int channel:(上位16ビット):0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI
int channel:(下位16ビット):チャンネル番号0-15
戻り値 0:正常終了
0以外:エラーコード

int zm_set_ch_output_level(char *out);
機能 各チャンネルの出力レベルの設定
引数 char *out:出力パラメータを記載した領域
戻り値 0:正常終了
0以外:エラーコード
備考 char *outからは以下の構成の出力パラメータを格納しておく。
out+0(short):デバイス(0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI)
out+2(short):チャンネル番号(0-15)
out+4(char):パラメータ有効ビット
(d0:移動速度,d1:開始レベル,d2:終了レベル)
対応したビット=1で、そのパラメータが後続する
out+5(short):移動速度(フェーダー移動速度。0で動かさない)
out+7(char):フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
out+8(char):フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

int zm_set_tr_output_level(char *out);
機能 各トラックの出力レベルの設定
引数 char *out:出力パラメータを記載した領域
戻り値 0:正常終了
0以外:エラーコード
備考 char *outからは以下の構成の出力パラメータを格納しておく。
out+0(short):トラック番号(0-65534)、
out+2(char):パラメータ有効ビット
(d0:移動速度,d1:開始レベル,d2:終了レベル)
対応ビット=1で、そのパラメータが後続する
out+3(short):移動速度(フェーダー移動速度。0で動かさない)
out+5(char):フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
out+6(char):フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

int zm_master_fader(char *fdr);
機能 マスターフェーダーの設定
引数 char *fdr:出力パラメータを記載した領域
戻り値 0:正常終了
0以外:エラーコード
備考 char *fdrからは以下の構成の出力パラメータを格納しておく。
out+0(short):デバイス(0=FM音源,1=AD PCM音源
$8000=MIDI1,$8001=MIDI2
$8002=MIDI3,$8003=MIDI4
-1=カレントMIDI)
out+2(char):パラメータ有効ビット
(d0:移動速度,d1:開始レベル,d2:終了レベル)
対応しビット=1でそのパラメータが後続する
out+3(short):移動速度(フェーダー移動速度。0で動かさない)
out+5(char):フェーダー移動開始レベル(0-128)(128が原音ﾚﾍﾞﾙ)
out+6(char):フェーダー移動終了レベル(0-128)(128が原音ﾚﾍﾞﾙ)

int zm_get_fader_status(void);
機能 フェーダーの状態取得
引数 なし
戻り値 d0,d1,d6,d7ビットに検査結果が反映される。
d0:実行中のチャンネルフェーダーすべてが目的に達している
(1:達成した 0:達成していない)
d1:実行中のマスターフェーダーすべてが目的に達している
(1:達成した 0:達成していない)
d6:チャンネルフェーダーの状態(1:実行中 0:実行されていない)
d7:マスターフェーダーの状態 (1:実行中 0:実行されていない)

int zm_get_play_time(void);
機能 演奏時間の取得
引数 なし
戻り値 16進表現で00HHMMSSという書式で返ってくる。
なおHH:0-99,MM:0-59,SS:0-59の値を取る。

char *zm_get_1st_comment(void);
機能 演奏中の曲のコメントを取り出す
引数 なし
戻り値 文字列のポインタ
終端コードは0

int zm_get_timer_mode(void);
機能 現在のテンポソースとなっているタイマの種類を返す
引数 なし
戻り値 0:タイマ A,1:タイマ B,2:YM3802タイマ

short *zm_get_track_table(void);
機能 音楽演奏トラックテーブルのアドレスを得る
引数 なし
戻り値 音楽演奏トラックテーブルのアドレス
備考 音楽演奏トラックテーブルには演奏に使用される
(short)トラック番号が(short)-1を終端コードとし羅列される。

short *zm_get_track_table_se(void);
機能 効果音演奏トラックテーブルのアドレスを得る
引数 なし
戻り値 効果音演奏トラックテーブルのアドレス
備考 効果音演奏トラックテーブルには演奏に使用される
(short)トラック番号が(short)-1を終端コードとし羅列される。

char *zm_get_play_work(int track);
機能 音楽演奏トラックワークのアドレスを得る
引数 int track:トラック番号(0-65534)
戻り値 音楽演奏トラックワークのアドレス
備考 音楽演奏トラックワークの内容と構成についてはMEASURE13を
参照のこと。

char *zm_get_play_work_se(int track);
機能 効果音演奏トラックワークのアドレスを得る
引数 int track:トラック番号(0-65534)
戻り値 効果音演奏トラックワークのアドレス
備考 効果音演奏トラックワークの内容構成は音楽演奏トラックと同一。

char *zm_get_buffer_information(void);
機能 バッファアドレステーブルのアドレス
引数 なし
戻り値 バッファアドレステーブルのアドレス
備考 バッファアドレステーブルの構成についてはMEASURE13を参照のこと

char *zm_zmsc_status(void);
機能 Z-MUSICの状態ワークのアドレス
引数 なし
戻り値 Z-MUSICの状態ワークのアドレス
備考 Z-MUSICの状態ワークのアドレスの構成についてはMEASURE13を参照のこと

int zm_calc_total(char *zmd, char **result);

機能 ZMDの演奏時間やトータルステップタイムなどの計算

引数 char *zmd:ZMDの先頭アドレス(ヘッダはあってもなくてもよい)
char **result:戻り値を格納するための変数

戻り値 0:正常終了
0以外:発生したエラーの数

備考 ・戻り値が0の場合はchar **resultには計算結果データ列を格納した領域のポインタが返される。計算結果データ列の構成についてはMEASURE10ファンクションZM_CALC_TOTALの項を参照のこと。
・戻り値が0以外の場合はchar **resultにはエラーの内容を格納したエラーテーブルのアドレスが返される。エラーテーブルの構成についてはMEASURE11を参照のこと。
・いずれの場合もchar **resultに返された領域は使用後zm_free_mem()などで解放すること。
・この関数はコンバイラに含まれる外部組み込みのファンクションとであるため、使用前にはコンパイラの常駐を確認する必要がある。

int zm_register_application(char *addr, int *result);

機能 外部アプリケーションの常駐解除ルーチンの登録

引数 char *addr:外部アプリケーションの常駐解除ルーチンの先頭アドレス
int *result:戻り値を格納するための変数

戻り値 0:エラー
0以外:正常終了

備考 Z-MUSICが常駐解除するときに同時に解除できるようにするための手続き。Z-MUSICが常駐解除するときchar *addrのルーチンにエントリする。
int *resultには登録IDが返される。このIDは常駐解除ルーチンの登録をキャンセルする(zm_unregister_application()参照)場合に用いる。
char *addrから始まる各解放ルーチンは必要処理を実行後、最後にZ-MUSIC側で解放してほしいメモリブロックがあるならばそのアドレスをa0.lに返すようにすること。なければa0.l=0を返すこと
char *addrからの解放ルーチンの実行後に、なにか表示したいメッセージが有る場合はその文字列のアドレスをa1.lに返すようにすること。なければa1.l=0を返すこと。
char *addrからの解放ルーチンでエラーが発生した場合は戻り値としてd0.l(0:no error(エラーなし),1:occupied(占有されている),-1:error(エラー発生))を返すことができる。エラーが発生した場合もその旨を告げるメッセージ文字列をa1.lに与えることができる。

int zm_unregister_application(int result);

機能 外部アプリケーションの常駐解除ルーチンの登録キャンセル

引数 int result:zm_register_application()の戻り値

戻り値 0:エラー
0以外:正常終了

int zm_release_driver(char **zmsc);

機能 Z-MUSICの解除

引数 char **zmsc:Z-MUSICの常駐解除アドレス

戻り値 0:正常終了
0以外:エラーコード(常駐解除失敗)

備考 char **zmscで示される領域をシステムのメモリ解放コールを呼び出して解放した時点でZ-MUSICの常駐解除が完全に行われたことになる。

int zm_occupy_zmusic(int mode);

機能 Z-MUSICの占有

引数 int mode:1=占有する,0=占有を解除する,-1=占有状態の検査

戻り値 mode=0,1のとき:以前の占有状態
(0=占有されていない,1=占有されている)
mode=-1のとき:現在の占有状態
(0=占有されていない,1=占有されている)

char *zm_hook_fnc_service(int fnc, char *job);

機能 Z-MUSICのファンクションコールのフック

引数 int fnc:ファンクション番号(0-127)
char *job:ユーザーファンクションのエントリアドレス
-1とすると設定は行わずint fncで指定したファンクション番号のファンクションのエントリアドレスのみを返す。

戻り値 フック前のファンクションエントリアドレス
0ならば空いていた。
-1はエラー

char *zm_hook_zmd_service(int zmd, char *job);

機能 ZMDオペレーションのフック

引数 int zmd(下位16ビット):ZMDコード(0-255)
int zmd(上位16ビット):(0:FM,1:AD PCM,2:MIDI,
-1:共通コマンド,
-2:効果音用共通コマンド,
-3:制御コマンド)
char *job:ユーザーZMDオペレーションのエントリアドレス
-1とすると設定は行わずint zmdで指定したZMDコードのオペレーションのエントリアドレスのみを返す。

戻り値 フック前のZMDオペレーションエントリアドレス
0ならば空いていた。
-1はエラー

int zm_occupy_compiler(int mode);

機能 コンパイラの占有

引数 int mode:1=占有,0=占有解除,-1=占有状態検査

戻り値 mode=0,1のとき:以前の占有状態(mode=-1のときを参照)
mode=-1のとき:0=占有されていない,1=占有されている

int zm_store_error(int err, int noferr, char **addr);

機能 エラーコードのストア

引数 int err(上位16):エラーの発生したファンクション番号(0-32767)
(下位16):エラーコード(0-65535)
int err=-1とした場合は問い合わせのみでストアは行わない
int noferr:エラーの合計個数値に幾つ加算するか
char **addr:エラーストックバッファの先頭アドレスを格納する変数

戻り値 現在ストックしているエラーの個数

備考 本関数はユーザーアプリケーション側で発生したエラーをZ-MUSIC側に知らせたい場合に使用する。ただしその場合、エラーコードフォーマットはMEASURE11で示してある形式に従わなければならない。
エラーコードに関してはMEASURE11を参照。

int zm_print_error(int mode, int lang, int noferr, char *ZMD, char *srcaddr, char *srcname, char *errtbl char **buff);

機能 エラーメッセージの表示

引数 int mode:メッセージの出力先(0:画面,1:バッファ)
int lang:メッセージの言語(0:英語,1:日本語)
int noferr:エラーの数
char *zmd:ZMDのファイル名(ない場合は0)
char *srcaddr:ソース(ZMS)のアドレス(ない場合は0)
char *srcname:ソース(ZMS)ファイル名(ない場合は0)
char *errtbl:エラーストックバッファの先頭アドレス
char **buff:出力バッファの先頭アドレスを格納する変数
(int mode=1のときのみ有効,バッファの終端には0)

戻り値 出力バッファのサイズ。ただし終端の0は含まない。

備考 エラーテーブルからエラーメッセージを得る場合などに使用する。
char *zmd〜char *srcnameは通常の外部アプリケーションならば0でよい。正しく設定すればタグジャンプに対応した出力を得ることができる。

char *zm_get_mem(int size, int emply);

機能 メモリの確保

引数 int size:確保サイズ

int emply:確保するメモリの用途ID
戻り値　0:確保に失敗した
0以外:確保したメモリブロックのアドレス
備考　メモリの用途IDはMEASURE10 ファンクションZM_GET_MEMの項を参照のこと。Z-MUSICに無関係のメモリブロックを確保する場合ならば当関数を用いる必要はない。

char *zm_enlarge_mem(int size, char *addr);

機能　メモリブロックのサイズの変更
引数　int size:変更後のサイズ
char *addr:zm_get_mem()で確保したメモリのアドレス
戻り値　確保しなおしたメモリブロックのアドレス
0ならば確保に失敗した事を意味する
備考　メモリブロックのサイズを大きくした場合などはメモリブロックの再配置が自動で行われる。そのためchar *addrで与えられたメモリブロックが別のアドレスに移動する場合がある。よってこの関数を実行後はchar *addrを戻り値のアドレス値に置き換えるべきである。
変更に失敗した場合は、char *addrのメモリブロックの内容は保証される。

int zm_free_mem(char *addr);

機能　メモリブロックの解放
引数　char *addr:zm_get_mem()で確保したメモリのアドレス
戻り値　0:解放成功
0以外:エラーコード

int zm_free_mem2(int emply);

機能　特定の用途IDを持ったメモリブロックの解放
引数　int emply:メモリの用途ID
戻り値　0:解放成功
0以外:エラーコード
備考　メモリの用途IDはMEASURE10 ファンクションZM_GET_MEMの項を参照のこと。

int zm_exchange_memid(int mode, int s_emply, int d_emply, char *addr);

機能　メモリブロックの用途IDの変更
引数　int mode:変更モード
(0:チェックなしで変更,1:チェックしてから更新
-1:int s_emplyの用途IDを持つメモリブロックをすべて
int d_emplyの用途IDへ変更する)
int s_emply:更新元メモリの用途ID
(int mode=0の場合は意味なし)
int d_emply:希望メモリの用途ID
char *addr:zm_get_mem()で確保したメモリのアドレス
戻り値　0:変更成功
0以外:エラーコード
備考　メモリの用途IDはMEASURE10 ファンクションZM_GET_MEMの項を参照のこと。

void zm_init_all(void)

機能　Z-MUSICの全初期化を行う
引数　なし
戻り値　Z-MUSICのバージョンID
備考　通常のアプリケーションは本関数を利用する必要はない。
バージョンIDはたとえばバージョン3.21ならば$00003241となる。
4の部分は固定値。

void zm_int_start(int tmtype);

機能　割り込み開始制御
引数　int tmtype:制御対象タイマタイプ
(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)
戻り値　なし
備考　通常のアプリケーションは本関数を利用する必要はない

void zm_int_stop(int tmtype);

機能　割り込み停止制御
引数　int tmtype:制御対象タイマタイプ
(-1:音楽タイマ,0:音楽+効果音タイマ,1:効果音タイマ)
戻り値　なし
備考　通常のアプリケーションは本関数を利用する必要はない

int zm_set_int_service(int clock, int tempo, char *entry);

機能　ユーザ割り込みサービスの設定
引数　int clock:全音符のクロック数(1-65535)
int tempo:テンポ(1-32767)
char *entry:ユーザー割り込みルーチンのエントリアドレス
戻り値　0:設定正常終了
0以外:設定失敗
備考　・割り込みルーチンの最後はRTEではならない。かならず全レジスタを保存したうえでRTSで終了帰還するような作りにすること。
・char *entry=-1とした場合はすでに登録されている割り込みサービスの割込み周期をint clock,int_tempoで変更する機能を果たす。

char *zm_check_int_service(void);

機能　ユーザー割り込みサービスの状態検査
引数　なし
戻り値　0:空いている
0以外:現在登録されているユーザー割込みルーチンのエントリアドレス

int zm_release_int_service(char *entry);

機能　ユーザー割り込みサービスの登録解除
引数　char *entry:登録したユーザー割り込みルーチンのエントリアドレス
戻り値　0:登録解除正常終了
0以外:登録解除失敗

int zm_control_tempo(int mode);

機能　テンポ設定の主導権設定
引数　int mode:-1=テンポ設定の主導権を外部アプリケーションに委託
0=テンポ設定の主導権をZ-MUSICに戻す
(このとき内部パラメータでテンポの初期化も実施する)
戻り値　0:設定正常終了
0以外:設定失敗

void zm_convert_pcm(int mode, int size, char *source, char *destination);

機能　PCMデータの変換を行う
引数　int mode:変換モード(0:AD PCM to PCM,1:PCM TO AD PCM)
int size:変換元の(AD)PCMのデータサイズ
char *source:変換元となる(AD)PCMデータの格納領域アドレス
char *destination:変換先となる(AD)PCMデータの格納領域アドレス
戻り値　なし

int zm_work(int track, int offset);

機能　演奏トラックワークの内容取得
引数　int track:トラック番号(0-65534)
int offset:演奏トラックワークオフセット
戻り値　ワーク内容(0～255)
備考　演奏トラックワークの内容と構成についてはMEASURE13を参照のこと。

int zm_exec_zmd(int track, int size, char *zmd);

機能　ZMD列の実行
引数　int track:トラック番号(0-65534,65535:特に指定しない場合)
int size:ZMD列の長さ
char *zmd:ZMD列格納バッファ
(バッファ自体はd2.1+8の大きさを必要とする)
戻り値　なし

Z-MUSIC用語辞典

ここで紹介している用語は、一般的なシンセサイザ用語の意味とは異なったZ-MUSICに特化した説明がなされている場合があります。

A

AD PCM(ADAPTIVE DIFFERENTIAL PULSE CODED MODULATION)
X680x0の内蔵音源のひとつで、自然音をそのままデジタルレコーディングし、これを再生することができる。Z-MUSICではひとつの楽器音源としてコントロールが可能。

アフタータッチ(AFTER-TOUCH)
アフタータッチシーケンス(AFTER-TOUCH SEQUENCE)
鍵盤を押す強さを発音後に変化させること。MIDI楽器によっては、音色のニュアンスの変化を変えたり、ビブラートやトレモロなどの効果を与えたりすることができる。Z-MUSICでは発音後、音長の1/8単位でアフタータッチ処理することができる。この機能をZ-MUSICではアフタータッチシーケンスと呼ぶ。Z-MUSICではＦＭ音源に対してのアフタータッチもサポートしているが、この場合は音量の変化のみを制御することができる(MML @Z)。

アルゴリズム(ALGORITHM)
X680x0のFM音源は４つのオペレータ（発振器）で構成されているが、この各オペレータをどのように接続するかを決定するもの。X680x0に内蔵されているFM音源チップOPMは８つの「アルゴリズム」がプリセットされており、音色作成時にはこの８つから選ぶことになる。

アンプリチュードモジュレーション(AMPLITUDE MODULATION)
音量を周期的に変化させる音の表情付けの技法。トレモロともいわれる。Z-MUSICでは、４つのプリセット波形や、ユーザーが作成した波形に沿って変化させたり、発音後、音長の1/8ごとに掛かり具合を変化させたりすることができる。Z-MUSICではこの機能はFM音源専用となっているが、MIDI楽器ではARCCを音量に対して使うことで実現が可能。(MML @A)

ＡＲＣＣ(ASSIGNABLE REALTIME CONTROL CHANGE)
Z-MUSICにおける特殊機能のひとつ。各トラックごとに、コントロールするコントロールチェンジ(FM音源の場合は特定レジスタ)を選択し、これを音長の1/8ごとや設定した波形などに沿ってコントロールするもの。(MML @A)

アルペジオ(ARPPEGGIO)
分散和音。和音を構成する１音１音を少しずつ時間をずらして演奏するもの。Z-MUSICでは和音コマンドやダンパーコマンドで表現が可能。

オートベンド(AUTO BEND)
オートピッチベンド(AUTO PITCH BEND)
Z-MUSICの特殊機能のひとつで、発音後、任意の高さまで音程を滑らかに変化させることができる機能。(MML @B)

B

ベンド(BEND)→ピッチベンド(PITCH BEND)

C

キャリア(CARRIER)
FM音源のオペレータの名称で、主にエンベロープ/音量を決定する働きがある。

チャンネルプレッシャー(CHANNEL PRESSURE)
MIDIチャンネル単位のアフタータッチ。　→アフタータッチ

チェックサム(CHECK SUM)
一部のエクスクルーシブメッセージの最後尾につく、データを正確に受信できたかどうかを確かめるために用いられる検査用７ビットデータのこと。

コーラス(CHORUS)
エフェクタの一種で微妙に音程の違った音を重ねて音を厚くする機能を持つ。ローランドGS音源をはじめとしたDTM音源では標準で、このエフェクトを装備している。

クロック(CLOCK)
絶対音長のこと。Z-MUSICはデフォルトでは４分音符=48クロックで処理されている。

コントロールチェンジ(CONTROL CHANGE)
MIDI楽器には特殊な機能を手軽にコントロールするためにMIDI規格によって定められたレジスタ群があり、これをコントロールチェンジという。
(MML Y,@C)

D

ディレイ(DELAY)
ピッチモジュレーションをはじめとした、音に対して特殊な効果を与える際に、その効果が開始されるまでの遅延時間を決定するパラメータ。

デプス(DEPTH)→振幅

ディチューン(DETUNE)
音程を微調整するパラメータ。基準となる音高から決定される周波数とは違った音程で複数音を重ねて発音させ、コーラス効果を得たりする目的でよく用いられる。　(MML @K,@B)

デバイスID(DEVICE ID)
MIDI楽器に対してユーザーが設定した識別番号。

ダブルフラット(DOUBLE FLAT)
１度(２半音)下げる変化記号(MML --)

ダブルシャープ(DOUBLE SHARP)
１度(２半音)上げる変化記号(MML ++)

ダンパー(DAMPER)
ダンパーペダル(DAMPER PEDAL)
MIDIメッセージのひとつ。ダンパーオンにすると、以後、発音した音はノートオフの情報を無視し、発音をダンパーオフとするまで鳴り続ける。ダンパーペダルの状態設定はMIDIチャンネルごとに指定可能。Z-MUSICではFM音源やAD PCMにおいても使用可能となっている。(MML @D)

E

エフェクト(EFFECT)
エフェクタ(EFFECTOR)
音に特殊な味付けをすること。この装置をエフェクタという。MIDI楽器には音源のほかにエフェクタを持ったものが多くある。Z-MUSICではGS音源とローランドMT-32/CM系のエフェクタを手軽に操作する命令が備わっている。(MML @E)

エンベロープ(ENVELOPE)
音量、音程などのの時間的変化のこと。

エクスクルーシブメッセージ(EXCLUSIVE MESSAGE)
MIDI共通規格では定められていないパラメータを制御することができるMIDIメッセージ。MIDI楽器のメモリを直接書き換えたりすることができる。
(MML @X,X)

エクスプレッション(EXPRESSION)
そのMIDIチャンネルの出力音量の割合を決定するコントロールチェンジつ(コントロール番号11)。コントロールチェンジ７番のボリュームとは独立処理されるため、チャンネルごとの最終的な出力音量は

ボリューム(0〜127)×エクスプレッション(0〜127)

で決定される。

F

フラット(FLAT)
半音下げる変化記号。(MML -)

フェードイン(FADE IN)
音量を徐々に上げていくこと。(MML ¥)

フェードアウト(FADE OUT)
音量を徐々に下げていくこと。(MML ¥)

G

ゲートタイム(GATE TIME)
音符を実際に発音させる時間。

逆転再生→リバース

ＧＭ(GENERAL MIDI)
MIDI規格が推奨している、統一音色配列規格。

ＧＳ
ローランドが提唱したMIDI楽器の統一音色配列規格。GM規格を包括する。同社のSC-55やSC-88などがこの規格の代表的機種。

H

波形(WAVE TYPE)
Z-MUSICではピッチモジュレーション、アンプリチュードモジュレーション、拡張ARCCを、鋸歯状波、矩形波、三角波、鋸歯波シングル、ランダム波、ユーザー波形(波形メモリ)により制御することが可能である(MML S)。

波形メモリ(WAVE MEMORY)
Z-MUSICでは、ユーザーが波形の各ポイントの値を設定していき、非線形の波形を作ることができる。この機能を波形メモリ機能といい、Z-MUSICではピッチモジュレーション、アンプリチュードモジュレーション、拡張ARCCをこの機能でコントロールすることができる。
(ZMS .wave_form,MUSICZ3.FNC m_wave_form)

波長
波形の一周期の長さ。Z-MUSICではモジュレーション波形の波長を変更することができる。 (MML @S)

ホールド(HOLD)→ダンパー(DAMPER)

I

インストルメント(INSTRUMENT)
楽器のこと。

J K

拡張ピッチモジュレーション(EXTENDED PITCH MODULATION)
拡張ARCC(EXTENDED ARCC)
MIDIパートに対して有効な機能。FM音源パートのように5つのプリセット波形と波形メモリを用いたピッチモジュレーションやARCCが可能になる機能(MML M)。

キーオン(KEY ON)→NOTE ON

キーオフ(KEY OFF)→NOTE OFF

キートランスポーズ(KEY TRANSPOSE)
音高程を基準状態から上下にずらすこと。移調ともいう。同じような機能に「ディチューン(DETUNE)」がある。

L M

メーカーID(MAKER ID)
MIDI機器メーカーの識別番号。

MIDI(MUSICAL INSTRUMENT DIGITAL INTERFACE)
電子楽器の世界規格。メーカーや機種が違っていても、MIDI規格対応ならば同じコマンドメッセージで楽器をコントロールすることができる。

モデルＩＤ(MODEL ID)
MIDI機器メーカーが製品ごとに与えた識別番号。

モノ(MONO)
ひとつのMIDIチャンネルで一度に１音しか発音できないモード。トランペットなどのブラス系、シンセベースの音はこのモードで鳴らすと効果的。

モジュレータ(MODULATOR)
FM音源のオペレータの名称で、入力された波形を変調し音素を作り出す。

モジュール(MODULE)→音源モジュール

マルチティンバー(MULTI TIMBRE)
１台の音源で一度に複数の楽器音を演奏することができる機能のこと。

N

NRPN(NON REGISTERED PARAMETER NUMBER)
MIDI規格では定義されていないそのMIDI楽器特有の機能パラメータ。

ノートオン(NOTE ON)
発音すること。

ノートオフ(NOTE OFF)
発音した音を消音すること。

O

オクターブ(OCTAVE)
音高を8度単位で上下に移動させること。または、ある音階から8度離れている音のことをいう。 (MML O)

オムニ(OMNI)
MIDI機器において、設定された受信チャンネルとは無関係にすべてのチャンネル情報を認識するモードのこと。

音源モジュール
鍵盤のない音源のみで構成されたMIDI楽器のこと。コンピュータやシーケンサ、ほかのシンセサイザからMIDIメッセージによってコントロールすることによってのみ演奏を行うことができる。

音量→ボリューム(VOLUME)

音量モジュレーション→
アンプリチュードモジュレーション(AMPLITUDE MODULATION)

音程モジュレーション →ピッチモジュレーション(PITCH MODULATION)

ＯＰＭ
X680x0に内蔵されているヤマハ製のＦＭ音源チップの名称。型番はYM2151。4オペレータで構成され最大発音数は8。

P

パンポット(PANPOT)
発音する位置を設定するパラメータ。一般には定位という。FM音源やAD PCM音源では1,2,3がそれぞれ左、中央、右に対応する。MIDIではコントロールチェンジの10番がパンポットの機能に相当する。パラメータの範囲は0〜127までで、0がもっとも左、64が中央、127がもっとも右に対応する。
(MML P,@P)

パーシャル(PARTIAL)
音色を構成する発信源の名称。ひとつの音色が複数のパーシャルで構成されていることもある。主にローランド製のMIDI楽器で用いられる用語。

パーシャルリザーブ(PARTIAL RESERVE)
ローランドのマルチティンバー音源において、1パート内で保証されるパーシャル数を決定すること。一時的に音源に対して同時発音数を上回る発音要求があったとしてもパーシャルリザーブを行っていれば、その数のパーシャル数は確実にそのパートの発音のために保証される。

ピッチモジュレーション(PITCH MODULATION)
音程を周期的に変化させる音の表情付けの技法。ビブラートともいわれる。Z-MUSICでは、5つのプリセット波形や、ユーザーが作成した波形に沿って変化させたり、発音後、音長の1/8ごとに掛かり具合を変化させたりすることができる。

(MML @M)

ピッチベンド(PITCH BEND)
音程を滑らかに変化させること。Z-MUSICではパラメータを与えることでオートマチックにこの処理を行うことができる。(MML @B)

ポリ(POLY)
ひとつのMIDIチャンネルで一度に複数の音を発音できるモードのこと。

ポリフォニックプレッシャー(POLYPHONIC PRESSURE)
鍵盤単位で制御可能なアフタータッチ。 →アフタータッチ

ポルタメント(PORTAMENTO)
音程を滑らかに変化させること。 (MML (～),[～])

プログラムチェンジ(PROGRAM CHANGE)
音色切り替えのこと。

Q

クオンタイズ
リアルタイムレコーディングされた演奏情報をある規則に従って整理すること。

R

リバース(REVERSE)
(AD)PCMデータをデータ末尾から先頭へ向かって再生すること。Z-MUSICでは(AD)PCMコンフィギュレーションでこの指定が可能。

リバーブ(REVERB)
入力音に対して残響効果を与えて出力するエフェクタの一種。あたかもコンサートホールで演奏しているような効果を演出することができる。

ＲＰＮ(REGISTERED PARAMETER NUMBER)
MIDI規格で定義されたMIDI楽器の拡張機能パラメータ。ピッチベンドの幅のような、楽器の機種によらない一般的なパラメータが設定されている。

S

シャープ(SHARP)
半音上げる変化記号。(MML #)

振幅
ピッチモジュレーションなど、波形を用いて音に効果を与えるコマンドにおいて、その効果の掛かり具合を決定するパラメータ。(MML @M,@A)

ステップタイム(STEP TIME)
ひとつの音符に対してZ-MUSICが費やす時間。(MML L,@L)

ソステヌート(SOSTENUTO)
速さを抑え気味に、音を一様に持続して演奏する奏法。

スタンダードMIDIファイル(STANDARD MIDI FILE)
1988年アメリカ、オプコードシステムズ社により提唱された音楽データの標準形式として提唱されたファイルフォーマット。フォーマット0～2まであり、フォーマット0は1トラック構成、フォーマット1は複数トラック構成フォーマット2は複数トラックによる複数テンポ構成をとる。Z-MUSIC ver.3.0で扱えるのはフォーマット0のみ。フォーマット0でも最大16チャンネルのMIDI情報を記録できる。(MEASURE 9参照)

T

ティンバー(TIMBRE)
楽器(音色)のこと。Z-MUSIC ver.3.0では(AD)PCMパートにおいて、メロディ演奏を行いたい場合には、パートモードを「TIMBRE」に設定する必要がある。

(MML [PCM_MODE])

タイマ(TIMER)
Z-MUSICが音楽のテンポを管理するために使っているもの。X680x0のOPMにはタイマＡとタイマＢが存在し、Z-MUSICではどちらか一方のタイマを用いてテンポを制御可能になっている。タイマＢよりタイマＡのほうがテンポ決定の計算の際の誤差が少ない(MML T,@T)。

トーン(TONE)
音素のこと。Z-MUSIC ver.3.0では(AD)PCMパートにおいて、鍵盤1つ1つに対して(AD)PCMを割り当てて演奏するモードを「TONEモード」と呼んでいる。に設定する必要がある。

(MML [PCM_MODE])

トレモロ(TREMORO)
→アンプリチュードモジュレーション(AMPLITUDE MODULATION)

U

ユーザー波形→波形メモリ

V

ビブラート(VIBRATO)
→ピッチモジュレーション(PITCH MODULATION)

W

ウェイト(WAIT)
処理が終わるまで待ち時間を設けること。MIDI楽器ではEOX($F7)受信後、受信内容の処理のために多少のウェイトを必要とする。MMLのXコマンドでエクスクルーシブメッセージを送信する場合は、このウェイト期間を考慮して、命令後に休符などを設定したほうがよい。

X

ＸＧ
ヤマハが提唱したMIDI楽器の統一音色配列規格。GM規格を包括する。同社のMU50、MU80、MU90などがこの規格の代表的機種。

Y

Ｙコマンド
FM音源のレジスタを直接書き換えるＭＭＬ命令のこと。Z-MUSICではMIDIに対してのＹコマンドはコントロールチェンジとして扱われる。

Z

絶対音長(ABSOLUTE LENGTH)
Z-MUSICが処理する音長カウンタのこと。デフォルトでは4分音符は絶対音長48となっている(MML L,@L)。

マニュアル、およびZ-MUSICシステムに関するお問い合わせは

Z-MUSICのアフターサポートは下記の大手パソコン通信ネットワークで行っています。最新バージョンのアップロードも以下の2つのパソコン通信ネットワークで優先的に行っています。

意見、要望、バグレポートなども下記のパソコン通信ネットワークで取り扱っています。

●NECパソコン通信サービス PC-VAN

PC-VANにはX680x0ユーザーを中心として作られているSIG「X1CLUB」があります。

ログイン後

JX1CLUB

と打ち込むとX1CLUBに行くことができます。

X1CLUBにはZMUSIC専用の会議室があり、サポートはここで行っています。

X1CLUBのトップメニューから、

5.電子会議室

を選び、次に

4.会議　ZMUSIC会議室

を選びます。このあと

4.会議室へ入る

を選択するとZMUSIC会議室に入ることができます。

●NIFTY-Serve

NIFTY-ServeにはFSHARPUというシャープのパソコンユーザーのフォーラムがあります。トップメニューから、

GO FSHARPU

とするとFSHARPUフォーラムに行けます。

ここの「X68k 音楽」会議室でサポートを行っています。

●Oh!X関係ホームページ/Z-MUSIC関係ホームページ

インターネット接続できる方は、

http://www.softbank.co.jp/dosvstart/netx.htm

http://www.kt.rim.or.jp/~zenji/index.htm

で関連情報を入手することができます。

【注意】

Z-MUSICが意図した動作をしない場合は、必ず、その具体的な再現手順を明記してください。演奏データ(ZMS)などの演奏時に発生する不都合に関する報告は、症状を再現できるショートサンプルを作成して、レポートに添付するようにしてください。なお、改造マシンや、クロックアップ機、Z-MUSICと競合するソフトウェアを使用している環境下での不都合は対応しかねる場合があります。

サンプル曲について

今回は投稿作品からコナミ、カプコンのゲーム音楽を収録させていただきました。両社のライセンスによる曲目は以下のとおりです。

THUNDER CROSS全曲集
　(C) 1988 KONAMI ALL RIGHTS RESERVED.
MSX版GRADIUS 2全曲集
　(C) 1987 KONAMI ALL RIGHTS RESERVED.
GRADIUS II全曲集
　(C) 1988 KONAMI ALL RIGHTS RESERVED.
XEXEXより　Breeze
　(C) 1991 KONAMI ALL RIGHTS RESERVED.
究極戦隊ダダンダーンより　闘えダダンダーン
　(C) 1993 KONAMI ALL RIGHTS RESERVED.
Pop'n Twin Beeより TWIN FLIGHT ／ 虹色の風
　(C) 1995 KONAMI ALL RIGHTS RESERVED.
GRADIUS IIIよりFire Scramble／Cosmo Plant
　(C) 1989 KONAMI ALL RIGHTS RESERVED.
ファミコン版ドラキュラIIより街　The Silence of the Daylight
　(C) 1987 KONAMI ALL RIGHTS RESERVED.
バンパイアハンターより　モリガンのテーマ
　(C) CAPCOM ALL RIGHTS RESERVED.
ストリートファイター2　エンディングテーマ (Ryu)
　(C) CAPCOM ALL RIGHTS RESERVED.

MUSICZ3.FNCのアセンブルについて

手違いによりMUSICZ3.FNCの実行ファイルが収録されていませんでした。ソースリストは収録されていますので、アセンブルできる環境をお持ちの方はソースをアセンブル/リンクして、できあがったMUSICZ3.XをMUSICZ3.FNCにリネームしてお使いください。なお、アセンブルにはHAS.Xが必要になります。

また、CD-ROMに収録されているフリーソフトウェア群の中から、HAS.XとHLK.Xをbinなどのパスの通ったディレクトリに入れ、CD-ROMのZM3MSTRディレクトリにあるアーカイブZMSC3LIBを適当なディレクトリに展開し、このアーカイヴ同梱の、

AM.BAT

を実行すればアセンブルできます。

ただし、X68000ではそのままではアセンブルできないので、X68000ユーザーはコマンドラインから、

```
has /w musicz /s debug=0 -m68030
hlk musicz -o musicz3.fnc
```

とする必要があります。そして、CD,FD同梱のMUSICZ3.FNCはX68000ではMUSICZ.FNC(ZMUSIC ver.2.0のX-BASIC外部関数)のエミュレーションは実行できません。1月中旬、NIFTY,PC-VANなどにて公開予定の新バージョンではX68000/X68030両方でMUSICZ3.FNCのエミュレーションが実行できるようになっています。

Z-MUSICシステム ver.3.0

1997年1月15日　初版第1刷発行

元Oh!X編集部編

プログラム作成　西川善司

発行者　橋本五郎

発行所　ソフトバンク株式会社　出版事業部

〒103　東京都中央区日本橋箱崎町24-1

販売　03(5642)8101

編集　03(5642)8180

表紙CG　江口響子

印刷所　文唱堂印刷株式会社

落丁，乱丁本は小社販売局にてお取り替えいたします。
定価は裏表紙に表示してあります。

Printed in Japan 1997.

ISBN4-7973-0184-8 C0055

ISBN4-7973-0184-8
C0055 P6800E
9784797301847
1910055068001

定価6,800円
(本体6,602円)

元Oh!X編集部による奇跡の書

定価6,800円（本体6,602円）

Xユーザーの総力を結集！

ハードウェアパワーを200%絞り出す史上最強の音楽演奏記述言語！

- 65535トラック同時演奏可能。MIDI出力64チャンネルを実現
- AD PCM音源による16チャンネルメロディシーケンス機能搭載
- アゴーギク機能やエンハンストARCC機能を新装備
- 複数のX68000を接続して同期演奏可能
- 音階によって音色を切り換えるTIMBRE SPLIT機能搭載
- Z-MUSIC用サンプル曲100曲以上を収録
- CD-ROMにはXユーザーによる各種フリーウェアを満載

SOFT BANK